


Article

Fuzzy Automata as Coalgebras

Ai Liu ¹, Shun Wang ², Luis Soares Barbosa ³ and Meng Sun ^{2,*}

¹ Graduate School of Advanced Science and Engineering, Hiroshima University, Hiroshima 739-8511, Japan; liuai@hiroshima-u.ac.jp

² School of Mathematical Sciences, Peking University, Beijing 100871, China; wshun94@pku.edu.cn

³ INL (International Iberian Nanotechnology Laboratory) & INESC TEC, Universidade do Minho, 4704-553 Braga, Portugal; lsb@di.uminho.pt

* Correspondence: sunm@pku.edu.cn

Abstract: The coalgebraic method is of great significance to research in process algebra, modal logic, object-oriented design and component-based software engineering. In recent years, fuzzy control has been widely used in many fields, such as handwriting recognition and the control of robots or air conditioners. It is then an interesting topic to analyze the behavior of fuzzy automata from a coalgebraic point of view. This paper models different types of fuzzy automata as coalgebras with a monad structure capturing fuzzy behavior. Based on the coalgebraic models, we can define a notion of fuzzy language and consider several versions of bisimulation for fuzzy automata. A group of combinators is defined to compose fuzzy automata of two branches: state transition and output function. A case study illustrates the coalgebraic models proposed and their composition.

Keywords: fuzzy automata; coalgebra; fuzzy language; bisimulation; composition



Citation: Liu, A.; Wang, S.; Barbosa, L.S.; Sun, M. Fuzzy Automata as Coalgebras. *Mathematics* **2021**, *9*, 272. <https://doi.org/10.3390/math9030272>

Academic Editor: Tadashi Dohi

Received: 17 December 2020

Accepted: 25 January 2021

Published: 29 January 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Control logic plays an important role in component-based programming in deciding a run-time mechanisms and rules of composition. Precise control needs meticulous implementation so that many applications may be expensive and inefficient. To tackle this problem, there is an increasing interest in using fuzzy logic in many new areas. As a very efficient method for handling imprecise properties, fuzzy logic then provides a systematic approach to incorporating approximate reasoning into such systems so that fuzzy implementations are not only cheaper and faster than precise ones, but also more understandable for users [1,2]. Therefore, some devices that profit from the use of vagueness in their overall operation have emerged and the related theory is described in [3]. For instance, the fuzzy principal component analysis method, based on the variance contribution rate of the principal component combined with the fuzzy theory to obtain a reasonable correction weight, is used to refine quantitative and qualitative index data of innovation service capability [4]. Moreover, this approach makes sense not only at the control level, but also at the test level [5].

Fuzzy control systems incorporate a number of components driven by fuzzy logic [6]. Most of them are rule-based systems that exchange information through interfaces. Technically, the modeling approach of fuzzy control systems contains three aspects: an input stage, a processing stage and an output stage, whose details are as follows.

- The input stage transforms an input into a value. The method is to abstract the relation of an input and its corresponding vague value into a point in a coordinate system, where the horizontal axis stands for the input domain and the vertical axis stands for the vagueness domain.
- The processing stage involves inference rules and generates a result for each input, and then combines the results of the rule. In this stage, logical inference rules are used to describe the connection between cause and effect. The rules are of the form

If $\langle condition \rangle$ then $\langle conclusion \rangle$.

Such rules provide information for the decision of control variables.

- The output stage processes the combined results from the processing stage and converts them to a specific control value. For instance, common techniques for conversion process includes max-min inference, max-membership principle and mean-max membership.

Automata theory has a long history in modeling systems and applications which can be realized as a set of states and transitions between them depending on some inputs. Fuzzy finite-state automata (FFA) incorporate fuzziness into the internal state representation and output of these computational systems [7]. Depending on the non-fuzzy output labels associated with (final) states or transitions, there are different classes of FFA: FFA with final states, FFA without final states, Fuzzy Moore FA and Fuzzy Mealy FA [8]. There are also works considering fuzzy output maps, such as fuzzy Mealy machines and fuzzy Moore machines [9,10]. Fuzzy automata have been studied from different aspects. In order to study behavior control, a novel method to compute the membership values of the next states of a fuzzy automaton with an averaging function between the membership value of the input, and the membership value of the current state is proposed in [11]; the behaviors of lattice-valued nondeterministic fuzzy automata are compared through two language equivalence relations which have different discriminating power in [12]. Categories of deterministic fuzzy automata and fuzzy languages based on a complete residuated lattice with zero divisors are introduced in [13], a common framework for fuzzy type automata is developed in relationships with morphisms of monads in [14], and the concept of fuzzy regular language accepted by fuzzy finite automata is purposed in [15]. Describing systems that behave in the same way in the sense that one system simulates the other and vice versa, several notions of (approximate) bismulation relations are investigated in [16–18].

Along the past two decades, coalgebra has emerged as a well established general framework for the study of the behavior of various kinds of automata [19–21]. There is in particular a generalized determinization construction from automata to coalgebras, including partial Mealy machines, (structured) Moore automata, Rabin probabilistic automata, and pushdown automata [22]. A survey and hierarchy of probabilistic systems as coalgebras is discussed in [23]. It connects probabilistic verification with coalgebraic modeling and compares expressiveness of system types by natural transformations between functors. Hybrid automata specifying both discrete and continuous behavior can also be modeled as coalgebras [24]. A coalgebraic perspective supporting a generic theory of hybrid automata with a rich palette of definitions and results is studied in [25]. In addition, a coalgebraic semantics framework for quantum systems is developed in [26]. One obvious advantage of the coalgebraic view is that it induces a simple and intuitive notion of *bisimulation* between coalgebras, a notion originally stemming from the world of labeled transition systems and process algebra [27–29]. Witnessed by the notion of *coalgebra homomorphism*, bisimulation on coalgebras can be defined by commutative diagrams and shown to be formally dual to congruence on algebra [30,31]. Moreover, there is a general framework for the study of components as concrete coalgebras and the development of the corresponding calculi [32].

A recent thesis [33] proposes a coalgebraic approach to fuzzy automata, which obtains the following results: (a) a coalgebraic definition of the fuzzy language recognized by a fuzzy automaton, (b) the definition of a functor describing the determinization process of a fuzzy automata via a generalization of the powerset construction, (c) a coalgebraic definition of bisimulation on fuzzy automata allowing the construction of a quotient fuzzy automaton. However, it only considers the output as the current membership value for the current state. Moreover, a coalgebraic theory of fuzzy transition systems and their concrete fuzzy bisimulation is studied in [34]. The authors resort to relational lifting that is one of the most used methods in bisimulation research, leading to an algorithm for testing bisimulation in [35], and group-by-group fuzzy bisimulation and its corresponding modal logic in [36]. Nevertheless, the output stage is omitted. To consider different types of fuzzy automata, our main contributions are as follows:

- Explore the fuzzy-set monad to serve as the basis to a coalgebraic approach;
- Provide a coalgebraic framework for different types of fuzzy automata, where the notions of fuzzy language and bisimulation can be addressed;
- Define appropriate combinators for composing fuzzy automata from two branches: state transition and output function.

Thus, we not only consider fuzzy language respecting the controlling behavior and bisimulation relations for fuzzy automata, but also study the composition mechanism in our coalgebraic framework.

This paper is structured as follows. Section 2 introduces different types of fuzzy automata. Section 3 recalls the definition of the fuzzy-set monad and studies its properties. Section 4 defines the coalgebraic models for fuzzy automata, the notion of fuzzy language and considers several versions of bisimulation. Section 5 develops a series of combinators for composing fuzzy automata. Section 6 discusses a case study. Section 7 concludes and raises some topics for future work.

2. Fuzzy Automata

In a complex controlled system driven by fuzzy logic, a fuzzy automaton is the basic unit which contains fuzzy processors and input/output interfaces. Considering fuzzy output maps, we focus on three types of fuzzy automata: Fuzzy Moore Automata (FMrA), Fuzzy Mealy Automata (FMIA) and Fuzzy Unified Automata (FUA). FMrA and FMIA are obtained by modifying the definitions of fuzzy Moore machine and fuzzy Mealy machine in [8]. Unlike the definition of fuzzy Mealy machine in [8] requiring two functions, one to describe the next state and the other to describe the output, a fuzzy Mealy machine is equipped with one fuzzy function to characterize completely the next state and the output produced in [9]. For distinction between them, we name the latter one as FUA. For simplicity, initial and final states are ignored for the moment.

Definition 1 (Fuzzy Moore Automata (FMrA)). A fuzzy Moore automaton is a 5-tuple $p = (X, I, O, \alpha, e)$ where

- X is a set of states.
- I is a set of input symbols.
- O is a set of output symbols.
- $\alpha : X \times I \rightarrow [0, 1]^X$ is a fuzzy transition function.
- $e : X \rightarrow [0, 1]^O$ is a fuzzy output function.

Note that each non-fuzzy output map $e' : X \rightarrow O$ corresponds to a function $e : X \rightarrow [0, 1]^O$ such that $e(x) = \delta_{e'(x)}$, where $\delta_k(t) = \delta(t - k)$ and δ is the Dirac function.

Definition 2 (Fuzzy Mealy Automata (FMIA)). A fuzzy Mealy automaton is a 5-tuple $p = (X, I, O, \alpha, e)$ where

- $X, I, O, \alpha : X \times I \rightarrow [0, 1]^X$ are defined as in FMrA.
- $e : X \times I \rightarrow [0, 1]^O$ is a fuzzy input-output function.

Note that each non-fuzzy output map $e' : X \times I \rightarrow O$ corresponds to a fuzzy input-output function $e : X \times I \rightarrow [0, 1]^O$ where $e(x, i) = \delta_{e'(x, i)}$.

Given an FMrA (X, I, O, α, e) , it is easy to construct an FMIA (X, I, O, α, e') where $e'(x, i) = e(x)$ without loss of information, so we regard it as a subcase of FMIA and concentrate on the study of FMIA as coalgebras.

Definition 3 (Fuzzy Unified Automata (FUA)). A fuzzy unified automaton is a 4-tuple $p = (X, I, O, \beta)$ where

- X, I, O are defined as in FMrA.
- $\beta : X \times I \rightarrow [0, 1]^{X \times O}$ is a fuzzy input-transition-output function.

In classical methods, two operations $F_1 : [0, 1] \times [0, 1] \rightarrow [0, 1]$ and $F_2 : [0, 1]^* \rightarrow [0, 1]$ should be defined to define the language accepted by an automaton [7]. Instead, we intend to define the notion of fuzzy language with the aid of the fuzzy-set monad.

3. Fuzzy-Set Monad

3.1. Fuzzy Set

The fuzzy set theory [37] was developed by Lotfi A. Zadeh in 1965. The main purpose of using fuzzy sets is to deal with vague data under some given properties. For example, consider a finite set of real numbers $S \subseteq \mathbb{R}$ and the property “close to 0”. This property seems ambiguous because there is not an explicit criterion to judge whether objects are closed to 0. We want to ask within what distance we can say “one real number is close to 0”. To make it precise, the one should figure out a function which fits the property. For example,

$$\psi_S(x) = \max\{0, 1 - \frac{1}{m}|x|\}, \quad x \in [-m, m]$$

where $m = \max_{s \in S} |s|$. This function is called *the membership function* and indicates that the closer the data $s \in S$ is to 0, the closer the membership value $\psi_S(s)$ is to 0. Obviously, data from which the distance to 0 are equal have the same membership value, i.e., $\psi_S(s) = \psi_S(-s)$. However, the selection of membership function is not unique and usually depends on the goal of application.

Definition 4 (Residuated Lattice [33]). *A residuated lattice is an algebra $\mathcal{K} = (K, \wedge, \vee, \otimes, \rightarrow, 0, 1)$ with four binary and two nullary operations satisfying:*

- 1 $(K, \wedge, \vee, 0, 1)$ is a lattice with the partial order \leq which is defined by “ $x \leq y$ if and only if $x \vee y = y$ ”. The greatest (least) element is $1(0)$ that for all $x \in K$, $x \leq 1(x \geq 0)$;
- 2 $(K, \otimes, 1)$ is a commutative monoid with the unit element 1;
- 3 For $x, y, z \in K$, $x \leq y \rightarrow z$ if and only if $x \otimes y \leq z$.

Especially, if $(K, \wedge, \vee, 0, 1)$ is a complete lattice, then \mathcal{K} is called a complete residuated lattice.

Residuated lattices are the algebraic structure that characterizes fuzzy components.

Example 1. *The Boolean algebra $(\mathbf{2}, \wedge, \vee, \neg)$ is a residuated lattice $(\mathbf{2}, \wedge', \vee', \otimes', \rightarrow', 0, 1)$. In this expression, $\mathbf{2} = \{0, 1\}$ is the set of elements. \wedge', \vee' correspond to \wedge and \vee operations in Boolean algebra, respectively. Multiplication \otimes' is defined as \wedge . The residuate operation \rightarrow' comes as $x \rightarrow' y := \neg x \vee y$.*

Definition 5 (Fuzzy Subset [33]). *Given a set X , a fuzzy subset over \mathcal{K} of X is a function $\phi : X \rightarrow K$ that assigns to each object $x \in X$ a membership value. The set of all fuzzy subsets of X is denoted by $\mathcal{Z}_{\mathcal{K}}(X)$ and obviously $\mathcal{Z}_{\mathcal{K}}(X) = K^X$. In the sequel, we use the shorthand notation $\mathcal{Z}(X)$ to represent $\mathcal{Z}_{\mathcal{K}}(X)$.*

Note that \mathcal{Z} can be interpreted as an endofunctor on Set where

$$\begin{aligned} \mathcal{Z}(f) : K^X &\rightarrow K^Y \\ \kappa &\mapsto \lambda y. \bigvee_{x \in f^{-1}(y)} \kappa(x) \end{aligned}$$

for any $f : X \rightarrow Y$. Note that

$$\bigvee_{x \in X} \kappa(x) = \vee \{\kappa(x) | x \in X\}.$$

3.2. Properties of Fuzzy-Set Monad

The fuzzy-set monad on Set is defined in [33]. In this section, we will firstly recall the definition and then prove this monad is strong and commutative. Although every monad in Set is strong, we include the explicit contribution to build up intuitions.

Definition 6 (The fuzzy-set Monad [33]). *Fuzzy-set monad $\mathbb{Z} = (\mathcal{Z}, \eta, \mu)$ over $\mathcal{K} = (K, \wedge, \vee, \otimes, \rightarrow, 0, 1)$ satisfies for a set X*

- $\eta : Id \Rightarrow \mathcal{Z}$ satisfies that

$$\eta_X(x)(y) = \begin{cases} 1 & x = y \\ 0 & \text{otherwise,} \end{cases} \quad x, y \in X,$$

- $\mu : \mathcal{Z}^2 \Rightarrow \mathcal{Z}$ satisfies that

$$\mu_X(\Phi) = \bigcup_{\psi \in \mathcal{Z}(X)} \Phi(\psi) \otimes \psi, \quad \Phi \in \mathcal{Z}^2(X).$$

where

$$\left(\bigcup_{i \in I} \phi_i\right)(x) = \bigvee_{i \in I} \phi_i(x) \quad x \in X, \phi_i \in \mathcal{Z}(X)$$

and

$$(a \otimes \phi)(x) = a \otimes \phi(x) \quad a \in K, x \in X, \phi \in \mathcal{Z}(X)$$

Definition 7 (Strong monad [21]). *A strong monad is a monad $\mathbb{T} = (T, \eta, \mu)$ equipped with a left tensorial strength $\sigma_{X,Y} : T(X) \times Y \rightarrow T(X \times Y)$ that commutes with the unit and multiplication of the monad:*

$$\begin{array}{ccccc} X \times Y & \xleftarrow{\text{id}} & X \times Y & & T^2(X) \times Y \xrightarrow{\sigma_{T(X),Y}} T(T(X) \times Y) \xrightarrow{T(\sigma_{X,Y})} T^2(X \times Y) \\ \eta_X \times \text{id} \downarrow & & \downarrow \eta_{X \times Y} & \mu_X \times \text{id} \downarrow & \downarrow \mu_{X \times Y} \\ T(X) \times Y & \xrightarrow{\sigma_{X,Y}} & T(X \times Y) & & T(X) \times Y \xrightarrow{\sigma_{X,Y}} T(X \times Y) \end{array}$$

Theorem 1. *The triple $\mathbb{Z} = (\mathcal{Z}, \eta, \mu)$ is a strong monad.*

Proof. Firstly, define a left tensorial strength with components $\sigma_{X,Y} : \mathcal{Z}(X) \times Y \rightarrow \mathcal{Z}(X \times Y)$ as

$$\sigma_{X,Y}(\psi, y) = \lambda x. \lambda y'. (\psi(x) \otimes \eta_Y(y)(y'))$$

that commute appropriately with trivial projection and associativity isomorphisms for $f : X \rightarrow Z$ and $g : Y \rightarrow W$:

$$\begin{array}{ccc} \mathcal{Z}(X) \times 1 & \xrightarrow{\sigma_{X,1}} & \mathcal{Z}(X \times 1) \\ & \searrow \pi_1 & \downarrow \mathcal{Z}\pi_1 \\ & & \mathcal{Z}(X) \end{array} \qquad \begin{array}{ccc} \mathcal{Z}(X) \times Y & \xrightarrow{\sigma_{X,Y}} & \mathcal{Z}(X \times Y) \\ \mathcal{Z}(f) \times g \downarrow & & \downarrow \mathcal{Z}(f \times g) \\ \mathcal{Z}(Z) \times W & \xrightarrow{\sigma_{Z,W}} & \mathcal{Z}(Z \times W) \end{array}$$

$$\begin{array}{ccccc}
 (\mathcal{Z}(X) \times Y) \times Z & \xrightarrow{\sigma_{X,Y} \times \text{id}} & \mathcal{Z}(X \times Y) \times Z & \xrightarrow{\sigma_{X \times Y, Z}} & \mathcal{Z}((X \times Y) \times Z) \\
 \downarrow \cong & & & & \downarrow \cong \\
 \mathcal{Z}(X) \times (Y \times Z) & \xrightarrow{\sigma_{X, Y \times Z}} & & & \mathcal{Z}(X \times (Y \times Z))
 \end{array}$$

For the unit,

$$\begin{aligned}
 & \sigma_{X,Y} \cdot (\eta_X \times \text{id})(x, y) \\
 = & \quad \{ \text{Definition of } \times \quad \} \\
 & \sigma_{X,Y}(\eta_X(x), y) \\
 = & \quad \{ \text{Definition of } \sigma \quad \} \\
 & \otimes \cdot (\eta_X(x) \times \eta_Y(y)) \\
 = & \quad \{ \text{Definition of } \eta \quad \} \\
 & \eta_{X \times Y}(x, y).
 \end{aligned}$$

For the multiplication, we have to show that $\mu_{X \times Y} \cdot \mathcal{Z}(\sigma_{X,Y}) \cdot \sigma_{\mathcal{Z}(X), Y} = \sigma_{X,Y} \cdot (\mu_X \times \text{id})$.
 For a pair $(\Phi, y) \in \mathcal{Z}^2(X) \times Y$,

$$\begin{aligned}
 & \mu_{X \times Y} \cdot \mathcal{Z}(\sigma_{X,Y}) \cdot \sigma_{\mathcal{Z}(X), Y}(\Phi, y) \\
 = & \{ \text{Definition of } \sigma \} \\
 & \mu_{X \times Y} \cdot \mathcal{Z}(\sigma_{X,Y})(\otimes \cdot (\Phi \times \eta_Y(y))) \\
 = & \{ \text{Definition of } \mathcal{Z}, \sigma \} \\
 & \mu_{X \times Y} \left(\bigcup_{(\psi, y') \in \mathcal{Z}(X) \times Y} \otimes \cdot (\Phi \times \eta_Y(y))(\psi, y') \otimes \eta_{\mathcal{Z}(X \times Y)}(\sigma_{X,Y}(\psi, y')) \right) \\
 = & \{ \otimes \cdot (f \times g)(x, y) = f(x) \otimes g(y) \} \\
 & \mu_{X \times Y} \left(\bigcup_{(\psi, y') \in \mathcal{Z}(X) \times Y} (\Phi(\psi) \otimes \eta_Y(y)(y')) \otimes \eta_{\mathcal{Z}(X \times Y)}(\sigma_{X,Y}(\psi, y')) \right) \\
 = & \{ \text{Definition of } \eta \} \\
 & \mu_{X \times Y} \left(\bigcup_{\psi \in \mathcal{Z}(X)} \Phi(\psi) \otimes \eta_{\mathcal{Z}(X \times Y)}(\sigma_{X,Y}(\psi, y)) \right) \\
 = & \{ \text{Definition of } \sigma \} \\
 & \mu_{X \times Y} \left(\bigcup_{\psi \in \mathcal{Z}(X)} \Phi(\psi) \otimes \eta_{\mathcal{Z}(X \times Y)}(\otimes \cdot (\psi \times \eta_Y(y))) \right) \\
 = & \{ \text{Definition of } \mu \} \\
 & \bigcup_{\psi' \in \mathcal{Z}(X \times Y)} \left(\bigcup_{\psi \in \mathcal{Z}(X)} \Phi(\psi) \otimes \eta_{\mathcal{Z}(X \times Y)}(\otimes \cdot (\psi \times \eta_Y(y)))(\psi') \right) \otimes \psi' \\
 = & \{ \text{Definition of } \eta \} \\
 & \bigcup_{\psi \in \mathcal{Z}(X)} \Phi(\psi) \otimes (\otimes \cdot (\psi \times \eta_Y(y)))
 \end{aligned}$$

For the right side of the equation,

$$\begin{aligned}
 & \sigma_{X,Y} \cdot (\mu_X \times \text{id})(\Phi, y) \\
 &= \{ \text{Definition of } \mu \} \\
 & \sigma_{X,Y} \left(\bigcup_{\psi \in \mathcal{Z}(X)} \Phi(\psi) \otimes \psi, y \right) \\
 &= \{ \text{Definition of } \sigma \} \\
 & \otimes \cdot \left(\left(\bigcup_{\psi \in \mathcal{Z}(X)} \Phi(\psi) \otimes \psi \right) \times \eta_Y(y) \right) \\
 &= \{ \text{Distributive law: } \otimes \cdot (\cup_i f_i \times g) = \cup_i (\otimes \cdot (f_i \times g)) \} \\
 & \bigcup_{\psi \in \mathcal{Z}(X)} \otimes \cdot (\Phi(\psi) \otimes \psi \times \eta_Y(y)) \\
 &= \{ \text{Constant } \Phi(\psi) \} \\
 & \bigcup_{\psi \in \mathcal{Z}(X)} \Phi(\psi) \otimes (\otimes \cdot (\psi \times \eta_Y(y))).
 \end{aligned}$$

□

In the proof of Theorem 1, we defined a left tensorial strength σ with components $\sigma_{X,Y} : \mathcal{Z}(X) \times Y \rightarrow \mathcal{Z}(X \times Y)$ as

$$\sigma_{X,Y}(\psi, y) = \otimes \cdot (\psi, \eta_Y(y)) = \lambda x, \lambda y'. (\psi(x) \otimes \eta_Y(y)(y')).$$

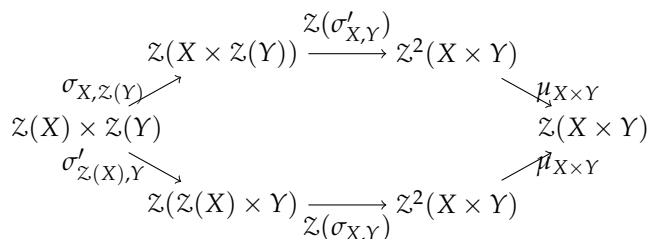
Of course, a “swapped” tensorial strength σ' with components $\sigma'_{X,Y} : X \times \mathcal{Z}(Y) \rightarrow \mathcal{Z}(X \times Y)$ can be obtained by applying swapping operation from the left tensorial strength:

$$(X \times \mathcal{Z}(Y) \xrightarrow[\cong]{s} \mathcal{Z}(Y) \times X \xrightarrow{\sigma_{Y,X}} \mathcal{Z}(Y \times X) \xrightarrow[\cong]{\mathcal{Z}(s)} \mathcal{Z}(X \times Y)).$$

where $s \hat{=} \langle \pi_2, \pi_1 \rangle$ is product communicating. Formally,

$$\sigma'_{X,Y} = \otimes \cdot (\eta_X(x) \times \phi) = \lambda x'. \lambda y. (\eta_X(x)(x') \otimes \phi(y)).$$

With both $\sigma_{X,Y}$ and $\sigma'_{X,Y}$, there are two ways to obtain $\mathcal{Z}(X) \times \mathcal{Z}(Y) \rightarrow \mathcal{Z}(X \times Y)$, as depicted in the following diagram. If the diagram commutes, then \mathcal{Z} is commutative with left and right strength natural transformations $\sigma_{X,Y}, \sigma'_{X,Y}$. We use $\gamma : \mathcal{Z}(X) \times \mathcal{Z}(Y) \rightarrow \mathcal{Z}(X \times Y)$ to denote the composed arrow.



Theorem 2. The triple (\mathcal{Z}, η, μ) is a commutative monad.

Proof. To show the diagram is commutative, select a pair of membership functions $(\psi_1, \psi_2) \in \mathcal{Z}(X) \times \mathcal{Z}(Y)$, then

$$\begin{aligned}
 & \mu_{X \times Y} \cdot \mathcal{Z}(\sigma'_{X,Y}) \cdot \sigma_{X,\mathcal{Z}(Y)}(\psi_1, \psi_2) \\
 = & \{ \text{Definition of } \sigma \} \\
 & \mu_{X \times Y}(\mathcal{Z}(\sigma'_{X,Y})(\otimes \cdot (\psi_1 \times \eta_{\mathcal{Z}(Y)}(\psi_2)))) \\
 = & \{ \text{Definition of } \mathcal{Z} \} \\
 & \mu_{X \times Y} \left(\bigcup_{(x,\psi) \in X \times \mathcal{Z}(Y)} \otimes \cdot (\psi_1 \times \eta_{\mathcal{Z}(Y)}(\psi_2))(x, \psi) \otimes \eta_{\mathcal{Z}(X \times Y)}(\sigma'_{X,Y}(x, \psi)) \right) \\
 = & \{ \otimes \cdot (f \times g)(x, y) = f(x) \otimes g(y) \} \\
 & \mu_{X \times Y} \left(\bigcup_{(x,\psi) \in X \times \mathcal{Z}(Y)} (\psi_1(x) \otimes \eta_{\mathcal{Z}(Y)}(\psi_2)(\psi)) \otimes \eta_{\mathcal{Z}(X \times Y)}(\sigma'_{X,Y}(x, \psi)) \right) \\
 = & \{ \text{Definition of } \eta \} \\
 & \mu_{X \times Y} \left(\bigcup_{x \in X} \psi_1(x) \otimes \eta_{\mathcal{Z}(X \times Y)}(\sigma'_{X,Y}(x, \psi_2)) \right) \\
 = & \{ \text{Definition of } \sigma' \} \\
 & \mu_{X \times Y} \left(\bigcup_{x \in X} \psi_1(x) \otimes \eta_{\mathcal{Z}(X \times Y)}(\otimes \cdot (\eta_X(x) \times \psi_2)) \right) \\
 = & \{ \text{Definition of } \mu \} \\
 & \bigcup_{\psi' \in \mathcal{Z}(X \times Y)} \left(\bigcup_{x \in X} \psi_1(x) \otimes \eta_{\mathcal{Z}(X \times Y)}(\otimes \cdot (\eta_X(x) \times \psi_2))(\psi') \otimes \psi' \right) \\
 = & \{ \text{Definition of } \eta \} \\
 & \bigcup_{x \in X} (\psi_1(x) \otimes (\otimes \cdot (\eta_X(x) \times \psi_2))) \\
 \cong & \{ \text{Denotation} \} \\
 & f_1
 \end{aligned}$$

For the right side of the equation,

$$\begin{aligned}
 & \mu_{X \times Y} \cdot \mathcal{Z}(\sigma_{X,Y}) \cdot \sigma'_{\mathcal{Z}(X),Y}(\psi_1, \psi_2) \\
 = & \{ \text{Definition of } \sigma \} \\
 & \mu_{X \times Y}(\mathcal{Z}(\sigma_{X,Y})(\otimes \cdot (\eta_{\mathcal{Z}(X)}(\psi_1) \times \psi_2))) \\
 = & \{ \text{Definition of } \mathcal{Z} \} \\
 & \mu_{X \times Y} \left(\bigcup_{(\psi,y) \in \mathcal{Z}(X) \times Y} \otimes \cdot (\eta_{\mathcal{Z}(X)}(\psi_1) \times \psi_2)(\psi, y) \otimes \eta_{\mathcal{Z}(X \times Y)}(\sigma_{X,Y}(\psi, y)) \right) \\
 = & \{ \otimes \cdot (f \times g)(x, y) = f(x) \otimes g(y) \} \\
 & \mu_{X \times Y} \left(\bigcup_{(\psi,y) \in \mathcal{Z}(X) \times Y} (\eta_{\mathcal{Z}(X)}(\psi_1)(\psi) \otimes \psi_2(y)) \otimes \eta_{\mathcal{Z}(X \times Y)}(\sigma_{X,Y}(\psi, y)) \right) \\
 = & \{ \text{Definition of } \eta \} \\
 & \mu_{X \times Y} \left(\bigcup_{y \in Y} \psi_2(y) \otimes \eta_{\mathcal{Z}(X \times Y)}(\sigma_{X,Y}(\psi_1, y)) \right) \\
 = & \{ \text{Definition of } \sigma \}
 \end{aligned}$$

$$\begin{aligned}
 & \mu_{X \times Y} \left(\bigcup_{y \in Y} \psi_2(y) \otimes \eta_{\mathbb{Z}(X \times Y)} (\otimes \cdot (\psi_1 \times \eta_Y(y))) \right) \\
 &= \{ \text{Definition of } \mu \} \\
 & \bigcup_{\psi' \in \mathbb{Z}(X \times Y)} \left(\bigcup_{y \in Y} \psi_2(y) \otimes \eta_{\mathbb{Z}(X \times Y)} (\otimes \cdot (\psi_1 \times \eta_Y(y))) (\psi') \otimes \psi' \right) \\
 &= \{ \text{Definition of } \eta \} \\
 & \bigcup_{y \in Y} (\psi_2(y) \otimes (\otimes \cdot (\psi_1 \times \eta_Y(y)))) \\
 &\hat{=} \{ \text{Denotation} \} \\
 & f_2
 \end{aligned}$$

Note that $f_1(x, y) = \psi_1(x) \otimes \psi_2(y) = \otimes \cdot (\psi_1 \times \psi_2)(x, y) = f_2(x, y)$. Hence the diagram commutes. \square

4. Going Coalgebraic

4.1. Coalgebraic Models

Since the automata introduced in Section 2 are defined over the interval $[0, 1]$, we assume the fuzzy-set monad $\mathbb{Z} = (\mathbb{Z}, \eta, \mu)$ is also defined over some complete residuated lattice $([0, 1], \min, \max, \otimes, \rightarrow, 0, 1)$. The corresponding coalgebraic models are based on the fuzzy-set monad.

Example 2 ([33]). Note that $([0, 1], \min, \max, 0, 1)$ is a complete lattice. Then there are several ways to construct complete residuated lattices $([0, 1], \min, \max, \otimes, \rightarrow, 0, 1)$; namely

- Define

$$\begin{aligned}
 x \otimes y &= \max(x + y - 1, 0) \\
 x \rightarrow y &= \min(1 - x + y, 1)
 \end{aligned}$$

for $x, y \in [0, 1]$. Then, $([0, 1], \min, \max, \otimes, \rightarrow, 0, 1)$ is a complete residuated lattice corresponding to the standard Lukasiewicz algebra.

- Define

$$\begin{aligned}
 x \otimes y &= \min(x + y - 1, 0) \\
 x \rightarrow y &= \begin{cases} 1 & \text{if } x \leq y \\ y & \text{if } y < x \end{cases}
 \end{aligned}$$

for $x, y \in [0, 1]$. Then, $([0, 1], \min, \max, \otimes, \rightarrow, 0, 1)$ is a complete residuated lattice corresponding to the standard Gödel algebra.

- Define

$$\begin{aligned}
 x \otimes y &= x \cdot y \\
 x \rightarrow y &= \begin{cases} 1 & \text{if } x \leq y \\ \frac{y}{x} & \text{if } y < x \end{cases}
 \end{aligned}$$

for $x, y \in [0, 1]$. Then, $([0, 1], \min, \max, \otimes, \rightarrow, 0, 1)$ is a complete residuated lattice corresponding to the standard product algebra.

Consider the two functors $F_{I,O} = \mathbb{Z}(- \times O)^I$ and $T_{I,O} = \mathbb{Z}(-)^I \times \mathbb{Z}(O)^I$. Given a FMIA (X, I, O, α, e) , the corresponding $T_{I,O}$ -coalgebra is $(X, \langle \bar{\alpha}, \bar{e} \rangle : X \rightarrow \mathbb{Z}(X)^I \times \mathbb{Z}(O)^I)$ where \bar{f} is the curried version of f . Given a FUA (X, I, O, β) , the corresponding $F_{I,O}$ -coalgebra is $(X, \bar{\beta} : X \rightarrow \mathbb{Z}(X \times O)^I)$. Obviously, there is a natural transformation θ from $T_{I,O}$ to $F_{I,O}$:

$$\theta(\langle f, g \rangle)(i) = \gamma(\langle f(i), g(i) \rangle)$$

for $f \in \mathcal{Z}(X)^I, g \in \mathcal{Z}(O)^I$ and $i \in I$. In the sequel, $F_{I,O}$ -coalgebras provide a universal framework for defining fuzzy language and bisimulation for different fuzzy automata while $T_{I,O}$ -coalgebras serve as a basis for composition calculi of fuzzy Mealy automata.

4.2. Fuzzy Language

In [33], fuzzy automata with initial fuzzy subsets and final fuzzy subsets are equipped with the notion of fuzzy language over a set of input symbols. Due to the type of their initial/final fuzzy subsets, that notion can not be naturally extended to the case involving output. Here we consider the notion of fuzzy language over a set of input symbols and a set of output symbols based on $F_{I,O}$ -coalgebras.

Definition 8. Let (X, f) be an $F_{I,O}$ -coalgebra. Define $f^* : X \rightarrow \mathcal{Z}(X \times O^*)^{I^*}$ as follows:

$$\begin{aligned} f^*(x)(i)(y, o) &= f(x)(i)(y, o) \\ f^*(x)(\emptyset)(y, \emptyset) &= \begin{cases} 1 & \text{if } x = y \\ 0 & \text{if } x \neq y \end{cases} \\ f^*(x)(i)(y, \emptyset) &= 0 \\ f^*(x)(\emptyset)(y, o) &= 0 \\ f^*(x)(wi)(y, vo) &= \bigvee_{z \in X} f^*(x)(w)(z, v) \otimes f^*(z)(i)(y, o) \end{aligned}$$

for $\forall x, y \in X, i \in I, o \in O, w \in I^*, v \in O^*$. Note that \emptyset represents the empty input/output.

Lemma 1. Given an $F_{I,O}$ -coalgebra (X, f) , $\forall x, y \in X, w \in I^*, v \in O^*$, if $|w| \neq |v|$ then

$$f^*(x)(w)(y, v) = 0.$$

Proof. First, we prove the result for $|w| > |v|$ by induction on $|w| = n$. Let $x, y \in X, w \in I^*, v \in O^*$. If $n = 0$, there exists no v such that $|v| < 0$ and hence the result holds. If $n = 1$, then $v = \emptyset$ and the result holds by the Definition 8. Assume that the result is true for all $|w| \in I^*$ such that $|w| = n - 1, n > 1$. Now there are two cases: $|v| = \emptyset$ and $|v| \neq \emptyset$. For the case $|v| = \emptyset$, let $|w| = w'i$, where $|w| = n, i \in I$, and then

$$f^*(x)(w'i)(y, \emptyset) = \bigvee_{z \in X} f^*(x)(w')(z, \emptyset) \otimes f^*(z)(i)(y, \emptyset).$$

By the induction hypothesis, $f^*(x)(w')(z, \emptyset) = f^*(z)(i)(y, \emptyset) = 0$ and thus the result holds. For the case $|v| \neq \emptyset$, let $w = w'i, v = v'o$ where $|w| = n > |y|, i \in I, o \in O$ and then

$$f^*(x)(w'i)(y, v'o) = \bigvee_{z \in X} f^*(x)(w')(z, v') \otimes f^*(z)(i)(y, o).$$

By the induction hypothesis, $f^*(x)(w')(z, v') = 0$ and hence $\forall z \in X, f^*(x)(w')(z, v') \otimes f^*(z)(i)(y, o) = 0$. Therefore, the result holds.

Second, by a similar proof, we can prove the result holds for $|w| < |v|$ by induction on $|y| = n$. \square

Lemma 2. Given an $F_{I,O}$ -coalgebra (X, f) , $\forall x, y \in X, w_1, w_2 \in I^*, v_1, v_2 \in O^*$, if $|w_1| = |v_1|$ and $|w_2| = |v_2|$, then

$$f^*(x)(w_1w_2)(y, v_1v_2) = \bigvee_{z \in X} f^*(x)(w_1)(z, v_1) \otimes f^*(z)(w_2)(y, v_2)$$

Proof. The results can be proved by induction on $|w_2| = n$. If $n = 0$, then $w_2 = v_2 = \emptyset$ and $w_1w_2 = w_1, v_1v_2 = v_1$. Since $f^*(x)(\emptyset)(y, \emptyset)$ is 1 when $x = y$ and $f^*(x)(\emptyset)(y, \emptyset)$ is 0 otherwise,

$$f^*(x)(w_1)(y, v_1) = \bigvee_{z \in X} f^*(x)(w_1)(z, v_1) \otimes f^*(z)(\emptyset)(y, \emptyset)$$

holds, which completes the proof of the base case. Now assume that the result is true for all $|w_2| = n - 1, n > 0$. Let $w_2 = w'i$ and $v_2 = v'o$, where $|w'| = |v'| = n - 1, i \in I, o \in O$. Then

$$\begin{aligned} & f^*(x)(w_1w_2)(y, v_1v_2) \\ &= f^*(x)(w_1w'i)(y, v_1v'o) \\ &= \bigvee_{z \in X} f^*(x)(w_1w')(z, v_1v') \otimes f^*(z)(i)(y, o) \\ &= \bigvee_{z \in X} \left(\bigvee_{r \in X} f^*(x)(w_1)(r, v_1) \otimes f^*(r)(w')(y, v') \right) \otimes f^*(z)(i)(y, o) \\ &= \bigvee_{z \in X} \left(\bigvee_{r \in X} f^*(x)(w_1)(r, v_1) \otimes f^*(r)(w')(y, v') \right) \otimes f^*(z)(i)(y, o) \\ &= \bigvee_{r \in X} \left(\bigvee_{z \in X} f^*(x)(w_1)(r, v_1) \otimes f^*(r)(w')(y, v') \otimes f^*(z)(i)(y, o) \right) \\ &= \bigvee_{r \in X} \left(f^*(x)(w_1)(r, v_1) \otimes \bigvee_{z \in X} f^*(r)(w')(y, v') \otimes f^*(z)(i)(y, o) \right) \\ &= \bigvee_{r \in X} f^*(x)(w_1)(r, v_1) \otimes f^*(r)(w'i)(y, v'o) \\ &= \bigvee_{r \in X} f^*(x)(w_1)(r, v_1) \otimes f^*(r)(w_2)(y, v_2) \end{aligned}$$

□

Now we consider a generic fuzzy language for $F_{I,O}$ -coalgebras and naturally obtain the definition for the fuzzy language accepted by a fuzzy automaton.

Definition 9 (Fuzzy language). A fuzzy language over an input set I and an output set O (with membership values over \mathcal{K}), is a fuzzy subset of $(IO)^*$, that is a function $\phi : (IO)^* \rightarrow [0, 1]$.

Example 3. For instance, let $I = \{i_1, i_2\}, O = \{o_1, o_2\}$. A fuzzy language ϕ can be defined as $\phi(i_1o_1) = 0.6, \phi(i_1o_2) = 0.8, \phi(i_2o_1) = 0.5, \phi(i_2o_2) = 1$ and $\phi(s) = 0, \forall s \in (IO)^*, |s| \neq 2$.

Definition 10. Consider an $F_{I,O}$ -coalgebra $(X, f : X \rightarrow \mathcal{Z}(X \times O)^I)$. For $w = i_1o_1i_2o_2 \dots \in (IO)^*$, define $w_i = i_1i_2 \dots$ and $w_o = o_1o_2 \dots$. Given an initial fuzzy state $\epsilon \in \mathcal{Z}(X)$ and a final fuzzy state $\tau \in \mathcal{Z}(X)$, the fuzzy language L_f recognized by (X, f) is defined by

$$L_f(w) = \bigvee_{x, y \in X} \epsilon(x) \otimes f^*(x)(w_i)(y, w_o) \otimes \tau(y), \quad w \in (IO)^*.$$

Naturally, the fuzzy language recognized by a FUA (X, I, O, β) is the one recognized by its corresponding $F_{I,O}$ -coalgebra $(X, \bar{\beta})$.

When considering the language recognized by an FMIA, the membership values of the next state and the output must be integrated, which can be captured by the natural transformation θ .

Definition 11. The fuzzy language recognized by a FMIA (X, I, O, α, e) is the one recognized by the corresponding $F_{I,O}$ -coalgebra $(X, \theta(\bar{\alpha}, \bar{e}))$.

4.3. Bisimulation

Let us now discuss the notion of bisimulation for fuzzy automata. In fact, coalgebra theory provides a generic notion of bisimulation on H -coalgebras for any functor H [20].

Definition 12 (H -bisimulation). *Given two H -coalgebras $(X, f : X \rightarrow H(X))$ and $(Y, g : Y \rightarrow H(Y))$, an H -bisimulation between them is a relation $R \subseteq X \times Y$ such that there exists an H -coalgebra $(R, h : R \rightarrow H(R))$ making the following diagram to commute.*

$$\begin{array}{ccccc}
 X & \xleftarrow{\pi_1} & R & \xrightarrow{\pi_2} & Y \\
 f \downarrow & & \downarrow h & & \downarrow g \\
 H(X) & \xleftarrow{H(\pi_1)} & H(R) & \xrightarrow{H(\pi_2)} & H(Y)
 \end{array}$$

Theorem 3. *Given two $T_{I,O}$ -coalgebras (X, f) and (Y, g) , if $R \subseteq X \times Y$ is a $T_{I,O}$ -bisimulation, then R is an $F_{I,O}$ -bisimulation between $(X, \theta \circ f)$ and $(Y, \theta \circ g)$.*

Proof. The proof of the result is immediate from the definition. \square

We now consider concrete bisimulations for different types of fuzzy automata. Since FMra can be easily transformed to FMIA, we only focus on bisimulation for FMIA and FUA.

Given a FMIA (X, I, O, α, e) , denote a transition $x \xrightarrow[o, v_2]{i, v_1} x'$ if $\alpha(x, i)(x') = v_1, e(x, i)(o) = v_2$.

Given a FUA (X, I, O, β) , denote a transition $x \xrightarrow{i|v|o} x'$ if $\beta(x, i)(x', o) = v$.

Definition 13 (Bisimulation for FMIA). *Given two FMIA (X, I, O, α, e) and (Y, I, O, α', e') , $R \subseteq X \times Y$ is a concrete bisimulation if it satisfies the following properties.*

- For $(x, y) \in R$, if $x \xrightarrow[o, v_2]{i, v_1} x'$, there exists $y' \in Y$, such that $y \xrightarrow[o, v_2]{i, v_1} y'$ and $(x', y') \in R$.
- For $(x, y) \in R$, if $y \xrightarrow[o, v_2]{i, v_1} y'$, there exists $x' \in X$, such that $x \xrightarrow[o, v_2]{i, v_1} x'$ and $(x', y') \in R$.

Definition 14 (Bisimulation for FUA). *Given two FUA $p = (X, I, O, \beta)$ and $q = (Y, I, O, \beta')$, $R \subseteq X \times Y$ is a concrete bisimulation if it satisfies the following properties.*

- For $(x, y) \in R$, if $x \xrightarrow{i|v|o} x'$, there exists $y' \in Y$, such that $y \xrightarrow{i|v|o} y'$ and $(x', y') \in R$.
- For $(x, y) \in R$, if $y \xrightarrow{i|v|o} y'$, there exists $x' \in X$, such that $x \xrightarrow{i|v|o} x'$ and $(x', y') \in R$.

Theorem 4. *Given two FMIA (X, I, O, α, e) and (Y, I, O, α', e') , R is a concrete bisimulation if and only if R is a $T_{I,O}$ -bisimulation between their corresponding $T_{I,O}$ -coalgebras.*

Proof. The proof of the result is immediate from the definition. \square

Theorem 5. *Given two FUA (X, I, O, β) and (Y, I, O, β') , R is a concrete bisimulation if and only if R is an $F_{I,O}$ -bisimulation between their corresponding $F_{I,O}$ -coalgebras.*

Proof. The proof of the result is immediate from the definition. \square

Since the core idea of fuzzy automata is fuzzing, the concrete bisimulation induced by coalgebraic bisimulation seems to be too strict. To find a more suitable characterization of bisimulation of fuzzy automata, we introduce the notion of approximate ϵ -bisimulation, which requires that membership values for states in an approximate ϵ -bisimulation of two transition branches should have a difference less than ϵ .

Definition 15 (ϵ -Bisimulation for FMIA). Given two FMIA (X, I, O, α, e) and (Y, I, O, α', e') , a relation $R \subseteq X \times Y$ is an approximate ϵ -bisimulation ($\epsilon > 0$) if for all $(x, y) \in R$:

- If $x \xrightarrow{o, v_2}^{i, v_1} x'$, there exists $y' \in Y$, such that $y \xrightarrow{o, u_2}^{i, u_1} y'$, $|u_1 - v_1| \leq \epsilon, |u_2 - v_2| \leq \epsilon$ and $(x', y') \in R$.
- If $y \xrightarrow{o, u_2}^{i, u_1} y'$, there exists $x' \in X$, such that $x \xrightarrow{o, v_2}^{i, v_1} x'$, $|u_1 - v_1| \leq \epsilon, |u_2 - v_2| \leq \epsilon$ and $(x', y') \in R$.

Example 4. Consider two FMIA (X, I, O, α, e) and (Y, I, O, α', e') , where $X = \{x_1, x_2\}, Y = \{y_1, y_2\}, I = \{i\}, O = \{o\}, \alpha(x_1, i)(x_2) = 0.6, e(x_1, i)(o) = 0.4, \alpha'(y_1, i)(y_2) = 0.5, e'(y_1, i)(o) = 0.5$. Then, $R = \{(x_1, y_1), (x_2, y_2)\}$ is an approximate 0.1-bisimulation.

Definition 16 (ϵ -Bisimulation for FUA). Given two FUA (X, I, O, β) and (Y, I, O, β') , a relation $R \subseteq X \times Y$ is an approximate ϵ -bisimulation ($\epsilon > 0$) if for all $(x, y) \in R$,

- If $x \xrightarrow{i|u|o} x'$, then there exists y' such that $y \xrightarrow{i|v|o} y', |u - v| \leq \epsilon$ and $(x', y') \in R$;
- If $y \xrightarrow{i|v|o} y'$, then there exists x' such that $x \xrightarrow{i|u|o} x', |u - v| \leq \epsilon$ and $(x', y') \in R$.

Example 5. Consider two FUA (X, I, O, β) and (Y, I, O, β') , where $X = \{x_1, x_2\}, Y = \{y_1, y_2\}, I = \{i\}, O = \{o\}, \beta(x_1, i)(x_2, o) = 0.8, \beta'(y_1, i)(y_2, o) = 0.7$. Then, $R = \{(x_1, y_1), (x_2, y_2)\}$ is an approximate 0.1-bisimulation.

Proposition 1. For approximate ϵ -bisimulation, we have

1. R is an approximate ϵ -bisimulation if and only if R^{-1} is an approximate ϵ -bisimulation.
2. If R_i is an approximate ϵ_i -bisimulation for $i = 1, 2$, then $R_1 \circ R_2$ is an approximate $(\epsilon_1 + \epsilon_2)$ -bisimulation.
3. If R_i is an approximate ϵ_i -bisimulation, then $\cup_i R_i$ is an approximate $\max_i \{\epsilon_i\}$ -bisimulation.

Proof. The proof of the result is immediate from the definition. \square

5. Composition for FMIA

A family of combinators for $B(- \times O)^I$ -coalgebras where B is a monad, such as sequential composition ($;$), parallel (\boxtimes), choice (\boxplus) and concurrency (\boxdot) combinators were introduced in [32]. Therefore, the composition of FUA can be naturally instantiated. However, an FMIA assigns different membership values to the next state and the corresponding output, which should be separated for composition. With some abuse of notation, we construct sequential composition ($;$), parallel (\boxtimes), choice (\boxplus) and concurrency (\boxdot) combinators for FMIA. Consider three fuzzy Mealy automata p, q, r with the corresponding coalgebras

$$\begin{aligned}
 \llbracket p \rrbracket &= (X_p, \langle \bar{\alpha}_p, \bar{e}_p \rangle : X_p \rightarrow \mathcal{Z}(X_p)^I \times \mathcal{Z}(O)^I) \\
 \llbracket q \rrbracket &= (X_q, \langle \bar{\alpha}_q, \bar{e}_q \rangle : X_q \rightarrow \mathcal{Z}(X_q)^I \times \mathcal{Z}(R)^I) \\
 \llbracket r \rrbracket &= (X_r, \langle \bar{\alpha}_r, \bar{e}_r \rangle : X_r \rightarrow \mathcal{Z}(X_r)^O \times \mathcal{Z}(R)^O).
 \end{aligned}
 \tag{*}$$

Some standard isomorphisms in Set are used in the definitions of combinators:

$$\begin{aligned}
 a &: A \times B \times C \rightarrow A \times (B \times C) \\
 s &: A \times B \rightarrow B \times A \\
 xr &: A \times B \times C \rightarrow A \times C \times B \\
 m &: A \times B \times (C \times D) \rightarrow A \times C \times (B \times D) \\
 dist &: A \times (B + C) \rightarrow A \times B + A \times C
 \end{aligned}$$

Furthermore, combinators a_+, s_+, xr_+, m_+ are the corresponding isomorphisms for sums in Set. Finally, the inverse of an isomorphism i is denoted by i^{-1} .

The sequential composition combinator $;$ requires the compatibility of interfaces. The sequential composition of p, r actually shares the data which is sent out from p . From a coalgebraic point of view, it is a $T_{I,R}$ -coalgebra

$$\llbracket p; r \rrbracket = (X_p \times X_r, \langle \overline{\alpha_{p;r}}, \overline{e_{p;r}} \rangle)$$

where $\alpha_{p;r}$ is defined as:

$$\begin{aligned} X_p \times X_r \times I &\xrightarrow{xr} X_p \times I \times X_r \xrightarrow{\langle \alpha_p, e_p \rangle \times id} \mathcal{Z}(X_p) \times \mathcal{Z}(O) \times X_r \xrightarrow{a \circ xr} \mathcal{Z}(X_p) \times (X_r \times \mathcal{Z}(O)) \\ &\xrightarrow{id \times \sigma'_{X_r, O}} \mathcal{Z}(X_p) \times \mathcal{Z}(X_r \times O) \xrightarrow{id \times \mathcal{Z}\alpha_r} \mathcal{Z}(X_p) \times \mathcal{Z}\mathcal{Z}(X_r) \xrightarrow{\gamma \circ (id \times \mu)} \mathcal{Z}(X_p \times X_r) \end{aligned}$$

and $e_{p;r}$ is defined as:

$$X_p \times X_r \times I \xrightarrow{xr} X_p \times I \times X_r \xrightarrow{e_p \times id} \mathcal{Z}(O) \times X_r \xrightarrow{\sigma'_{X_r, O} \circ s} \mathcal{Z}(X_r \times O) \xrightarrow{\mathcal{Z}e_r} \mathcal{Z}\mathcal{Z}(R) \xrightarrow{\mu} \mathcal{Z}(R)$$

The parallel combinator \boxtimes corresponds to synchronous product and composes two coalgebras into one with their inputs (outputs) merged together. The parallel $p \boxtimes q$ produces an output belonging to $O \times R$ after receiving an input belonging to $I \times J$. Coalgebraically, the semantics of the parallel combinator is a $T_{I \times J, O \times R}$ -coalgebra

$$\llbracket p \boxtimes q \rrbracket = (X_p \times X_q, \langle \overline{\alpha_{p \boxtimes q}}, \overline{e_{p \boxtimes q}} \rangle)$$

where $\alpha_{p \boxtimes q}$ is defined as:

$$X_p \times X_q \times (I \times J) \xrightarrow{m} X_p \times I \times (X_q \times J) \xrightarrow{\alpha_p \times \alpha_q} \mathcal{Z}(X_p) \times \mathcal{Z}(X_q) \xrightarrow{\gamma} \mathcal{Z}(X_p \times X_q)$$

and $e_{p \boxtimes q}$ is defined as

$$X_p \times X_q \times (I \times J) \xrightarrow{m} X_p \times I \times (X_q \times J) \xrightarrow{e_p \times e_q} \mathcal{Z}(O) \times \mathcal{Z}(R) \xrightarrow{\gamma} \mathcal{Z}(O \times R)$$

The choice $p \boxplus q$ allows the environment to choose either to input a value of type I or one of type J , which will trigger the corresponding automata, producing the associated output. A formal definition is

$$\llbracket p \boxplus q \rrbracket = (X_p \times X_q, \langle \overline{\alpha_{p \boxplus q}}, \overline{e_{p \boxplus q}} \rangle)$$

where $\alpha_{p \boxplus q}$ is defined as

$$\begin{aligned} X_p \times X_q \times (I + J) &\xrightarrow{dist} X_p \times X_q \times I + X_p \times X_q \times J \xrightarrow{xr+a} X_p \times I \times X_q + X_p \times (X_q \times J) \\ &\xrightarrow{\alpha_p \times id + id \times \alpha_q} \mathcal{Z}(X_p) \times X_q + X_p \times \mathcal{Z}(X_q) \xrightarrow{[\sigma_{X_p, X_q}, \sigma'_{X_p, X_q}]} \mathcal{Z}(X_p \times X_q) \end{aligned}$$

and $e_{p \boxplus q}$ is defined as

$$\begin{aligned} X_p \times X_q \times (I + J) &\xrightarrow{dist} X_p \times X_q \times I + X_p \times X_q \times J \xrightarrow{xr+a} X_p \times I \times X_q + X_p \times (X_q \times J) \\ &\xrightarrow{e_p \circ \pi_1 + e_q \circ \pi_2} \mathcal{Z}(O) + \mathcal{Z}(R) \xrightarrow{[\mathcal{Z}(t_1), \mathcal{Z}(t_2)]} \mathcal{Z}(O + R) \end{aligned}$$

The concurrency combinator \boxtimes combines choice and parallel, in the sense that two fuzzy Mealy automata p and q can be executed depending on the input supplied. Let $I \boxtimes J = I + J + I \times J$ and $O \boxtimes R = O + R + O \times R$. The semantics of \boxtimes is given by

$$\llbracket p \boxtimes q \rrbracket = (X_p \times X_q, \langle \overline{\alpha_{p \boxtimes q}}, \overline{e_{p \boxtimes q}} \rangle)$$

where $\alpha_{p \boxtimes q}$ is defined as

$$\begin{aligned} X_p \times X_q \times (I \boxtimes J) &\xrightarrow{\text{dist}} X_p \times X_q \times (I + J) + X_p \times X_q \times (I \times J) \\ &\xrightarrow{\alpha_{p \boxplus q} + \alpha_{p \boxtimes q}} \mathcal{Z}(X_p \times X_q) + \mathcal{Z}(X_p \times X_q) \xrightarrow{[\mathcal{Z}(\text{id}), \mathcal{Z}(\text{id})]} \mathcal{Z}(X_p \times X_q) \end{aligned}$$

and $e_{p \boxtimes q}$ is defined as

$$\begin{aligned} X_p \times X_q \times (I \boxtimes J) &\xrightarrow{\text{dist}} X_p \times X_q \times (I + J) + X_p \times X_q \times (I \times J) \\ &\xrightarrow{e_{p \boxplus q} + e_{p \boxtimes q}} \mathcal{Z}(O + R) + \mathcal{Z}(O \times R) \xrightarrow{[\mathcal{Z}(t_1), \mathcal{Z}(t_2)]} \mathcal{Z}(O \boxtimes R) \end{aligned}$$

In coalgebra theory, it is [20] shown that the graph of a $T_{I,O}$ -homomorphism is a $T_{I,O}$ -bisimulation and the greatest $T_{I,O}$ -bisimulation is an equivalence relationship \sim . Thus for two given FMIA p, q , if there exists a $T_{I,O}$ -homomorphism between their corresponding coalgebras $\llbracket p \rrbracket, \llbracket q \rrbracket$, we denote $p \sim q$.

Theorem 6. For appropriately typed FMIA p, q, r, p', q' ,

$$\begin{aligned} (p; q); r &\sim p; (q; r) \\ (p \boxtimes p'); (q \boxtimes q') &\sim (p; q) \boxtimes (p', q') \\ (p \boxplus p'); (q \boxplus q') &\sim (p; q) \boxplus (p', q') \\ (p \boxtimes p'); (q \boxtimes q') &\sim (p; q) \boxtimes (p', q') \end{aligned}$$

Proof. The proof proceeds by pointwise induction. For the first law, if we assume

$$\begin{aligned} \alpha_p(x_1, i)(x'_1) &= k_1, e_p(x_1, i)(j) = t_1 \\ \alpha_q(x_2, j)(x'_2) &= k_2, e_q(x_2, j)(o) = t_2 \\ \alpha_r(x_3, o)(x'_3) &= k_3, e_r(x_3, o)(h) = t_3 \end{aligned}$$

we can obtain

$$\begin{aligned} &\alpha_{(p;q);r}(x_1, x_2, x_3)(i)(x'_1, x'_2, x'_3) \\ &= k_1 \otimes k_2 \otimes k_3 \otimes t_1 \otimes t_2 \\ &= \alpha_{p;(q;r)}(x_1, (x_2, x_3))(i)(x'_1, (x'_2, x'_3)) \\ &\quad e_{(p;q);r}(x_1, x_2, x_3)(i)(h) \\ &= t_1 \otimes t_2 \otimes t_3 \\ &= e_{p;(q;r)}(x_1, (x_2, x_3))(i)(h) \end{aligned}$$

With these equations, it is easy to show α is a $T_{I,O}$ -homomorphism from $\llbracket (p; q); r \rrbracket$ to $\llbracket p; (q; r) \rrbracket$. Other laws can be proved similarly. \square

Connecting FMIA through isomorphisms leads to a bisimilarity up to an isomorphic rearranging of input types and output types. Let f, g be isomorphic rearrangements of input types and output types respectively. We use $p\{f, g\}$ to denote the FMIA after arranging the input and the output types in the FMIA p .

Theorem 7. For appropriately typed FMLA p, q, r ,

$$\begin{aligned}
 p \boxtimes q &\sim (q \boxtimes p) \{s, s\} \\
 p \boxplus q &\sim q \boxplus p \{s_+, s_+\} \\
 p \boxdot q &\sim q \boxdot p \{s_+ + s, s_+ + s\} \\
 (p \boxtimes q) \boxtimes r &\sim p \boxtimes (q \boxtimes r) \{a, a^{-1}\} \\
 (p \boxplus q) \boxplus r &\sim p \boxplus (q \boxplus r) \{a_+, a_+^{-1}\} \\
 (p \boxdot q) \boxdot r &\sim p \boxdot (q \boxdot r) \{a_*, a_*^{-1}\}
 \end{aligned}$$

where a_* is a natural isomorphism from $(A \boxdot B) \boxdot C$ to $A \boxdot (B \boxdot C)$ and its inverse is denoted by a_*^{-1} .

Proof. Similar to Theorem 6. \square

The two theorems demonstrate that our combinators are well defined. In the sequel, we compare them with the ones in [32] up to the natural transformation θ through a theorem and an example.

Theorem 8. Given two FMLA p, q with the corresponding coalgebras in (\star) , the following equations holds.

$$\begin{aligned}
 \theta(\langle \overline{\alpha_{p \boxtimes q}}, \overline{e_{p \boxtimes q}} \rangle) &= \theta(\langle \overline{\alpha_p}, \overline{e_p} \rangle) \boxtimes \theta(\langle \overline{\alpha_q}, \overline{e_q} \rangle) \\
 \theta(\langle \overline{\alpha_{p \boxplus q}}, \overline{e_{p \boxplus q}} \rangle) &= \theta(\langle \overline{\alpha_p}, \overline{e_p} \rangle) \boxplus \theta(\langle \overline{\alpha_q}, \overline{e_q} \rangle) \\
 \theta(\langle \overline{\alpha_{p \boxdot q}}, \overline{e_{p \boxdot q}} \rangle) &= \theta(\langle \overline{\alpha_p}, \overline{e_p} \rangle) \boxdot \theta(\langle \overline{\alpha_q}, \overline{e_q} \rangle)
 \end{aligned}$$

where $\boxtimes, \boxplus, \boxdot$ correspond to our combinators in the left side and the ones for composing $F_{1,O}$ -coalgebras in [32] in the right side.

Proof. The proof proceeds by pointwise induction. For the first law, if we assume

$$\begin{aligned}
 \alpha_p(x_1, i)(x'_1) &= k_1, e_p(x_1, i)(j) = t_1 \\
 \alpha_q(x_2, j)(x'_2) &= k_2, e_q(x_2, j)(o) = t_2
 \end{aligned}$$

we obtain

$$\begin{aligned}
 &\theta(\langle \overline{\alpha_{p \boxtimes q}}, \overline{e_{p \boxtimes q}} \rangle)((x_1, x_2), i)((x'_1, x'_2), o) \\
 &= \overline{\alpha_{p \boxtimes q}}((x_1, x_2), i)(x'_1, x'_2) \otimes \overline{e_{p \boxtimes q}}((x_1, x_2), i)(o) \\
 &= (k_1 \otimes k_2) \otimes (t_1 \otimes t_2) \\
 &= (k_1 \otimes t_1) \otimes (k_2 \otimes t_2) \\
 &= \theta(\langle \overline{\alpha_p}, \overline{e_p} \rangle)(x_1, i)(x'_1, o) \otimes \theta(\langle \overline{\alpha_q}, \overline{e_q} \rangle)(x_2, j)(x'_2, o) \\
 &= \theta(\langle \overline{\alpha_p}, \overline{e_p} \rangle) \boxtimes \theta(\langle \overline{\alpha_q}, \overline{e_q} \rangle)
 \end{aligned}$$

Other laws can be proved similarly. \square

Note that the case for the sequential composition combinator does not always hold. Actually, this depends on the complete residuated lattice used, since the state transition of the first component is considered twice, which can be demonstrated by the following example.

Example 6. Recall the standard product algebra in Example 2. Consider two FMLA $p = (\{x_1, x_2\}, \{a\}, \{b\}, \alpha_p, e_p)$ and $r = (\{y_1, y_2\}, \{b\}, \{c\}, \alpha_r, e_r)$ where $\alpha_p(x_1, a)(x_2) = 0.4$, $e_p(x_1, a)(b) = 0.5$ and $\alpha_r(y_1, b)(y_2) = 0.8$, $e_r(y_1, b)(c) = 0.5$. Then we can obtain $\llbracket p; r \rrbracket = (U, \langle \overline{\alpha_{p;r}}, \overline{e_{p;r}} \rangle)$ where $U = \{(x_i, y_j) | i, j = 1, 2\}$, $\alpha_{p;r}((x_1, y_1), a)(x_2, y_2) = 0.4 \times 0.5 \times 0.8 = 0.16$ and $e_{p;r}((x_1, y_1), a)(c) = 0.5 \times 0.5 = 0.25$. Therefore

$$\theta(\langle \overline{\alpha_{p;r}}, \overline{e_{p;r}} \rangle)((x_1, y_1), a)((x_2, y_2), c) = 0.16 \times 0.25 = 0.04.$$

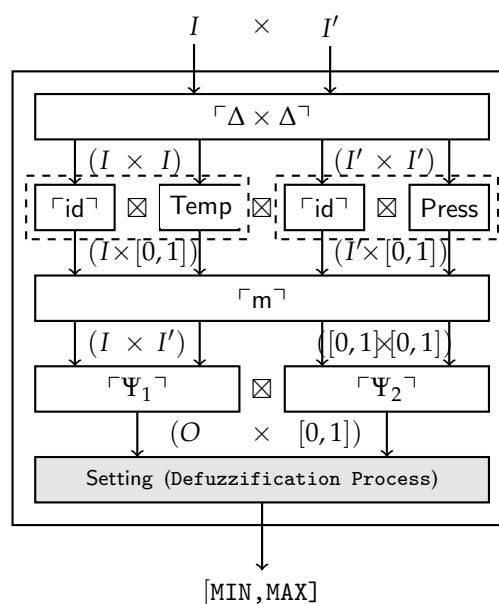
However, $\theta(\langle \bar{\alpha}_p, \bar{e}_p \rangle)(x_1, a)(x_2, b) = 0.4 \times 0.5 = 0.2$ and $\theta(\langle \bar{\alpha}_r, \bar{e}_r \rangle)(x_1, a)(x_2, b) = 0.8 \times 0.5 = 0.4$. Thus,

$$\theta(\langle \bar{\alpha}_p, \bar{e}_p \rangle); \theta(\langle \bar{\alpha}_r, \bar{e}_r \rangle)((x_1, y_1), a)((x_2, y_2), c) = 0.2 \times 0.4 = 0.08.$$

Oppositely, if we consider the standard Gödel algebra, the two values will be both 0.4.

6. Case Study

In the sequel, we illustrate the use of fuzzy components by means of a concrete example. For simplicity, we consider an non-fuzzy input-output function and compose components with $F_{I,O}$ -coalgebras. Consider the following example of a steam turbine.



The system is composed of two fuzzification components Temp, Press and a defuzzification component Setting with corresponding membership functions illustrated in Figure 1. Note that Δ represents the copy operation.

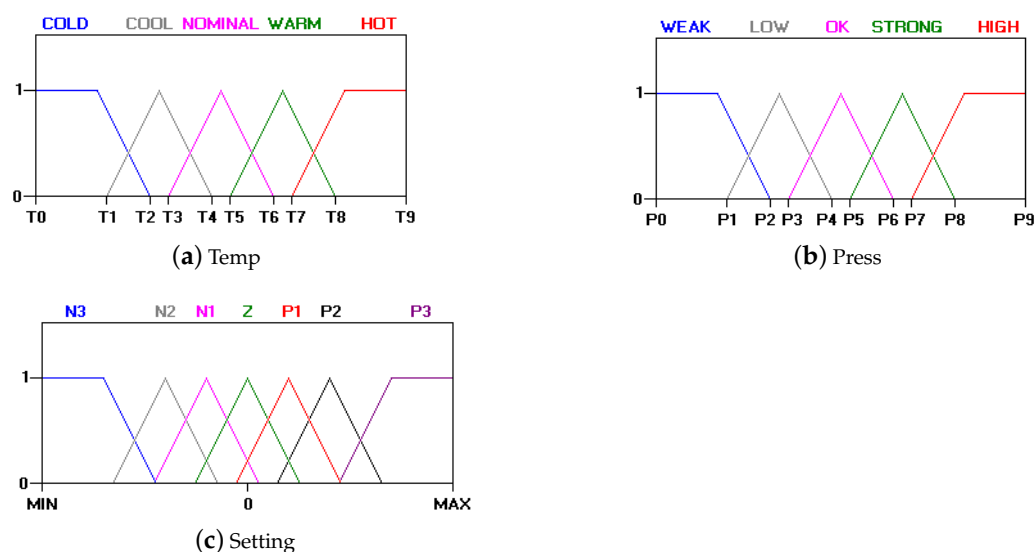


Figure 1. The graphs of membership functions.

In practice, the components Temp and Press execute in parallel. Each one will produce a membership value corresponding to the state and membership function after receiving a

mode signal. After that, the minimum of the two output values will become the input of Setting. The membership function of the Setting component is determined by the following rules (for simplicity, only whose conditions with temperature COOL are displayed).

- rule 1: If temperature is COOL and pressure is WEAK then throttle is P3.
- rule 2: If temperature is COOL and pressure is LOW then throttle is P2.
- rule 3: If temperature is COOL and pressure is OK then throttle is Z.
- rule 4: If temperature is COOL and pressure is STRONG then throttle is N2.
- rule 5: If temperature is COOL and pressure is HIGH then throttle is N3.
- ...

The output functions are considered as non-fuzzy in this example.

- (i) The coalgebraic semantic of component Temp

$$\llbracket \text{Temp} \rrbracket = (T, \theta \langle \bar{\alpha}_t, \bar{e}_t \rangle) : T \rightarrow \mathcal{Z}(T \times [0, 1])^I$$

is actually an $F_{I, [0, 1]}$ -coalgebra. In this model, states are the temperature over $T = [T_0, T_9]$, inputs are operation modes over set $I = \{\text{COLD}, \text{COOL}, \text{NORMAL}, \text{WARM}, \text{HOT}\}$ that are decided by users. The fuzzy transition function is constant on the temperature and given by $\alpha_t : T \times I \rightarrow [0, 1]^T$ with

$$\langle t, \text{COLD} \rangle \mapsto \phi_{\text{COL}}, \quad \langle t, \text{COOL} \rangle \mapsto \phi_{\text{COO}}, \quad \langle t, \text{NORMAL} \rangle \mapsto \phi_{\text{NOR}}, \quad \langle t, \text{WARM} \rangle \mapsto \phi_{\text{WAR}}, \quad \langle t, \text{HOT} \rangle \mapsto \phi_{\text{HOT}}$$

for all $t \in [T_0, T_9] \subseteq \mathbb{R}$. The output function $e_t : T \times I \rightarrow [0, 1]$ is defined by $\langle t, i \rangle \mapsto \text{eval}(\alpha_t(t, i), t)$ where eval is an evaluation function. As a concrete example, suppose the fuzzy subset for the NORMAL mode is the function

$$\phi_{\text{NOR}}(t) = \max\left\{0, \frac{2}{T_3 - T_6} \left(t - \frac{T_3 + T_6}{2}\right) + 1\right\}.$$

Then the membership value (output) over state $\frac{T_3 + T_6}{2}$ under the mode NORMAL is $e_t(\frac{T_3 + T_6}{2}, \text{NORMAL}) = \text{eval}(\phi_{\text{NOR}}, \frac{T_3 + T_6}{2}) = 1$.

- (ii) Press is a component whose state space P is given by the pressure in the steam turbine and inputs are over the set $I' = \{\text{WEAK}, \text{LOW}, \text{OK}, \text{STRONG}, \text{HIGH}\}$, which represent the mode triggered by the users. The output of this component is the membership value corresponding to the current fuzzy state. The dynamics of this component is

$$\llbracket \text{Press} \rrbracket = (P, \theta \langle \bar{\alpha}_p, \bar{e}_p \rangle) : P \rightarrow \mathcal{Z}(P \times [0, 1])^{I'}$$

with the transition and output functions defined as $\alpha_p : P \times I' \rightarrow \mathcal{Z}(P)$:

$$\langle p, \text{WEAK} \rangle \mapsto \phi_{\text{WEAK}}, \quad \langle p, \text{LOW} \rangle \mapsto \phi_{\text{LOW}}, \quad \langle p, \text{OK} \rangle \mapsto \phi_{\text{OK}}, \quad \langle p, \text{STRONG} \rangle \mapsto \phi_{\text{STRONG}}, \quad \langle p, \text{HIGH} \rangle \mapsto \phi_{\text{HIGH}}$$

for $p \in P$ and

$$o_p : P \times I' \rightarrow [0, 1] : (p, i') \mapsto \text{ev}(\alpha_p(p, i'), p).$$

- (iii) The dynamics of Rule and And components are denoted by $\ulcorner \Psi_1 \urcorner$ and $\ulcorner \Psi_2 \urcorner$ where

$$\ulcorner \Psi_1 \urcorner = (\mathbf{1}, \overline{\eta_{(1 \times O)} \cdot \langle \text{id}, \Psi_1 \rangle} : \mathbf{1} \rightarrow \mathcal{Z}(\mathbf{1} \times O)^{I \times I'}).$$

In this expression O is the output set determined by the output function, namely,

$$\Psi_1 : \mathbf{1} \times (I \times I') \rightarrow O$$

$$\Psi_1(*, (i, i')) = \begin{cases} \text{P3} & i = \text{COOL} \wedge i' = \text{WEAK} \\ \text{P2} & i' = \text{COOL} \wedge \text{LOW} \\ \text{Z} & i' = \text{COOL} \wedge \text{OK} \\ \text{N2} & i' = \text{COOL} \wedge \text{STRONG} \\ \text{N3} & i' = \text{COOL} \wedge \text{HIGH} \\ \dots & \end{cases}$$

$\mathbf{1} = \{*\}$ is the singleton set. The notation $\ulcorner f \urcorner$ is the representation of function $f : A \rightarrow B$, which is defined as a coalgebra $\ulcorner f \urcorner = (* \in \mathbf{1}, \bar{c}_{\ulcorner f \urcorner})$, where $c_{\ulcorner f \urcorner} = \mathbf{1} \times A \xrightarrow{\text{id} \times f} \mathbf{1} \times B \xrightarrow{\eta_{(\mathbf{1} \times B)}} \mathcal{Z}(\mathbf{1} \times B)$. The definition of Ψ_2 is similar, given a pair of inputs of $[0, 1]$, it outputs the minimum value of the two.

- (iv) The last component Setting works as follows. Through the channel it interacts with Temp and Press. It receives the mode information and a membership value as the current state. The mode information determines which membership function is accessible for the component. Then the component outputs an area whose boundary consists of the horizontal axis and the graph of the membership function. Formally, this model is represented by a coalgebra

$$\llbracket \text{Setting} \rrbracket = (D, \theta(\bar{\alpha}_s, \bar{e}_s)) : D \rightarrow \mathcal{Z}(D \times \mathcal{P}(\mathbb{R}^2))^{O \times [0,1]}$$

where $D = [\text{MIN}, \text{MAX}]$ is an interval of real numbers. The output function is defined as $e_s(d, (o, r)) = \{(x, y) \mid 0 \leq y \leq \min\{\alpha_s(x, (o, r)), r\}, x \in [\text{MIN}, \text{MAX}]\}$. Resorting to centroid defuzzification technique, the output stage processes combine areas and produce a control value, which will participate in the control of the system.

7. Conclusions and Future Work

The present work aims at addressing fuzzy automata from a coalgebraic perspective. Our starting point was studying the fuzzy-set monad further. We defined a left tensorial strength and a right tensorial strength, and proved it is a strong and commutative monad. With these properties, we modeled different types of fuzzy automata as coalgebraic models with the same transition structure. Based on these coalgebraic models, we defined the notions of fuzzy language bisimulation between fuzzy automata. Moreover, we developed some compositional combinators for fuzzy Mealy automata of two kinds: state transition and output function and compared it with the classical component calculi in [32]. Finally, through a case study, we discussed the application of our component calculi.

Besides these fundamental results, there are several topics left to explore. One is to define a notion of refinement [38] of fuzzy automata, to specify an inclusion relation of fuzzy behaviour. Fuzzy automata may involve complex behaviour such as non-deterministic transitions or branched transitions with probability [23,39]. Therefore another topic for future work is to develop more complex versions of fuzzy automata and analyze their behavior and discuss their properties, namely of the suitable notions of bisimulation as in [15,35,36].

Author Contributions: Conceptualization, M.S. and L.S.B.; methodology, A.L. and S.W.; formal analysis, A.L. and S.W.; investigation, A.L.; writing—original draft preparation, A.L. and S.W.; writing—review and editing, M.S. and L.S.B. All authors have read and agreed to the published version of the manuscript.

Funding: This work has been supported by the Guangdong Science and Technology Department (Grant No. 2018B010107004) and the National Natural Science Foundation of China under grant No. 61772038, 61532019 and 61272160. L.S.B. was supported by the ERDF—European Regional Development Fund through the Operational Programme for Competitiveness and Internationalisation—COMPETE 2020 Programme and by National Funds through the Portuguese funding agency, FCT, within project KLEE - POCI-01-0145-FEDER-030947.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: This work is also supported by Hiroshima University. Many thanks to the reviewers and editors.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Tanaka, K. *An Introduction to Fuzzy Logic for Practical Applications*; Springer: Berlin/Heidelberg, Germany, 1997.
2. Zadeh, L.A. Soft Computing and Fuzzy Logic. *IEEE Softw.* **1994**, *11*, 48–56. [[CrossRef](#)]
3. Syropoulos, A.; Grammenos, T. *Fuzzy Computation; A Modern Introduction to Fuzzy Mathematics*; John Wiley & Sons: Hoboken, NJ, USA, 2020; pp. 191–214. [[CrossRef](#)]
4. Wu, H.; Gu, X.; Zhen, L. Fuzzy Principal Component Analysis Model on Evaluating Innovation Service Capability. *Sci. Program.* **2020**, *2020*, 8834901. [[CrossRef](#)]
5. Böhme, M.; Pham, V.; Roychoudhury, A. Coverage-Based Greybox Fuzzing as Markov Chain. *IEEE Trans. Softw. Eng.* **2019**, *45*, 489–506. [[CrossRef](#)]
6. Simon, D.J. Introduction to Fuzzy Control. In *Embedded Systems Programming*; Electrical Engineering & Computer Science Faculty Publications: Cambridge, MA, USA, 2003; Volume 16, pp. 55–56.
7. Doostfateme, M.; Kremer, S.C. New directions in fuzzy automata. *Int. J. Approx. Reason.* **2005**, *38*, 175–214. [[CrossRef](#)]
8. Chaudhari, S.R.; Desai, A.S. On fuzzy Mealy and Moore machines. *Bull. Pure Appl. Math* **2010**, *4*, 375–384.
9. Mordeson, J.N.; Nair, P.S. Fuzzy Mealy machines. *Kybernetes* **1966**, *25*, 18–33. [[CrossRef](#)]
10. Li, Y.; Pedrycz, W. The equivalence between fuzzy Mealy and fuzzy Moore machines. *Soft Comput.* **2006**, *10*, 953–959. [[CrossRef](#)]
11. Todinca, D.; Sora, I.; Butoianu, D.; Precup, R. A Novel Method to Compute the Membership Value of the States of Fuzzy Automata. In Proceedings of the 2018 IEEE 12th International Symposium on Applied Computational Intelligence and Informatics (SACI), Timisoara, Romania, 17–19 May 2018; pp. 107–112. [[CrossRef](#)]
12. Pan, H.; Li, Y.; Cao, Y.; Li, P. Nondeterministic fuzzy automata with membership values in complete residuated lattices. *Int. J. Approx. Reason.* **2017**, *82*, 22–38. [[CrossRef](#)]
13. Tiwari, S.P.; Pal, P. On a category of deterministic fuzzy automata. In *11th Conference of the European Society for Fuzzy Logic and Technology (EUSFLAT 2019)*; Atlantis Studies in Uncertainty Modelling; Atlantis Press: Paris, France, 2019; Volume 1. [[CrossRef](#)]
14. Mockor, J. Monads and a common framework for fuzzy type automata. *Int. J. Gen. Syst.* **2019**, *48*, 406–442. [[CrossRef](#)]
15. Singh, A.K.; Tiwari, S.P. Fuzzy Regular Languages Based on Residuated Lattice. *New Math. Nat. Comput.* **2020**, *16*, 363–376. [[CrossRef](#)]
16. Yang, C.; Li, Y. Approximate bisimulations and state reduction of fuzzy automata under fuzzy similarity measures. *Fuzzy Sets Syst.* **2020**, *391*, 72–95. [[CrossRef](#)]
17. Yang, C.; Li, Y. ϵ -Bisimulation Relations for Fuzzy Automata. *IEEE Trans. Fuzzy Syst.* **2018**, *26*, 2017–2029. [[CrossRef](#)]
18. Yang, C.; Li, Y. Approximate bisimulation relations for fuzzy automata. *Soft Comput.* **2018**, *22*, 4535–4547. [[CrossRef](#)]
19. Rutten, J.J.M.M. Automata and coinduction (an exercise in coalgebra). In *International Conference on Concurrency Theory, Proceedings of the CONCUR 1998: CONCUR'98 Concurrency Theory, Nice, France, 8–11 September 1998*; Springer: Berlin/Heidelberg, Germany, 1998; Volume 1466, pp. 194–218.
20. Rutten, J.J.M.M. Universal coalgebra: A theory of systems. *Theor. Comput. Sci.* **2000**, *249*, 3–80. [[CrossRef](#)]
21. Jacobs, B. *Introduction to Coalgebra: Towards Mathematics of States and Observation*; Cambridge Tracts in Theoretical Computer Science; Cambridge University Press: Cambridge, UK, 2016; Volume 59.
22. Silva, A.; Bonchi, F.; Bonsangue, M.M.; Rutten, J.J.M.M. Generalizing determinization from automata to coalgebras. *Log. Methods Comput. Sci.* **2013**, *9*. [[CrossRef](#)]
23. Sokolova, A. Coalgebraic Analysis of Probabilistic Systems. Ph.D. Thesis, Technische Universiteit Eindhoven, Eindhoven, The Netherlands, 2005.
24. Neves, R.; Barbosa, L.S. Hybrid Automata as Coalgebras. In *International Colloquium on Theoretical Aspects of Computing, Proceedings of the ICTAC 2016: Theoretical Aspects of Computing, Taipei, Taiwan, 24–31 October 2016*; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2016; Volume 9965, pp. 385–402. [[CrossRef](#)]
25. Neves, R.; Barbosa, L.S. Languages and models for hybrid automata: A coalgebraic perspective. *Theor. Comput. Sci.* **2018**, *744*, 113–142. [[CrossRef](#)]
26. Liu, A.; Sun, M. A Coalgebraic Semantics Framework for Quantum Systems. In *International Conference on Formal Engineering Methods, Proceedings of the ICFEM 2019: Formal Methods and Software Engineering, Shenzhen, China, 5–9 November 2019*; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2019; Volume 11852, pp. 387–402. [[CrossRef](#)]
27. Feng, Y.; Duan, R.; Ying, M. Bisimulation for Quantum Processes. *ACM Trans. Program. Lang. Syst.* **2012**, *34*, 1–43. [[CrossRef](#)]
28. Larsen, K.G.; Skou, A. Bisimulation through probabilistic testing. *Inf. Comput.* **1991**, *94*, 1–28. [[CrossRef](#)]

29. Haghverdi, E.; Tabuada, P.; Pappas, G.J. Bisimulation Relations for Dynamical and Control Systems. *Electr. Notes Theor. Comput. Sci.* **2002**, *69*, 120–136. [[CrossRef](#)]
30. Jacobs, B. Invariants, Bisimulations and the Correctness of Coalgebraic Refinements. In *International Conference on Algebraic Methodology and Software Technology, Proceedings of the AMAST 1997: Algebraic Methodology and Software Technology, Sydney, Australia, 13–17 December 1997*; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 1997; Volume 1349, pp. 276–291. [[CrossRef](#)]
31. Venema, Y. Algebras and coalgebras. In *Handbook of Modal Logic; Studies in Logic and Practical Reasoning*; Elsevier B.V.: Amsterdam, The Netherlands, 2007; Volume 3, pp. 331–426. [[CrossRef](#)]
32. Barbosa, L.S. Components as Coalgebras. Ph.D. Thesis, Universidade do Minho, Braga, Portugal, 2001.
33. Guilherme, R.J.P. A Coalgebraic Approach to Fuzzy Automata. Ph.D. Thesis, Universidade Nova De Lisboa, Lisbon, Portugal, 2016.
34. Wu, H.; Chen, Y. Coalgebras for Fuzzy Transition Systems. *Electron. Notes Theor. Comput. Sci.* **2014**, *301*, 91–101. [[CrossRef](#)]
35. Wu, H.; Chen, Y.; Bu, T.; Deng, Y. Algorithmic and logical characterizations of bisimulations for non-deterministic fuzzy transition systems. *Fuzzy Sets Syst.* **2018**, *333*, 106–123. [[CrossRef](#)]
36. Wu, H.; Chen, T.; Han, T.; Chen, Y. Bisimulations for fuzzy transition systems revisited. *Int. J. Approx. Reason.* **2018**, *99*, 1–11. [[CrossRef](#)]
37. Nikraves, M.; Kacprzyk, J.; Zadeh, L.A. *Forging New Frontiers: Fuzzy Pioneers I*; University of California: Berkeley, CA, USA, 2007.
38. Meng, S.; Barbosa, L.S. Components as coalgebras: The refinement dimension. *Theor. Comput. Sci.* **2006**, *351*, 276–294. [[CrossRef](#)]
39. Narasimha, M.; Cleaveland, R.; Iyer, S.P. The role of observations in probabilistic open systems. *Electr. Notes Theor. Comput. Sci.* **1999**, *25*, 133–144. [[CrossRef](#)]