An integrative approach to simulation model discovery: Combining system theory, process mining and fuzzy logic

Yan Wang, Grégory Zacharewicz, Mamadou Kaba Traoré, David Chen

Journal of Intelligent & Fuzzy Systems xx (20xx) x–xx
DOI:10.3233/JIFS-17403
IOS Press

An integrative approach to simulation model discovery: Combining system theory, process mining and fuzzy logic

Yan Wang[a,*], Grégory Zacharewicz[a], Mamadou Kaba Traoré[b] and David Chen[a]
[a]*IMS, University of Bordeaux, Talence Cedex, France*
[b]*LIMOS, University of Blaise Pascal, Aubiere Cedex, France*

Abstract. System inference, i.e., the building of system structure from system behavior, is widely recognized as a critical challenging issue. In System Theory, structure and behavior are at the extreme sides of the hierarchy that defines knowledge about the system. System inference is known as climbing the hierarchy from less to more knowledge. In addition, it is possible only under justifying conditions. In this paper, a new system inference method is proposed. The proposed method extends the process mining technique to extract knowledge from data and to represent complex systems. The modularity, frequency and timing aspects can be extracted from the data. They are integrated together to construct the Fuzzy Discrete Event System Specification (Fuzzy-DEVS) model. The proposed approach consists of three stages: (1) extraction of event logs from data by using the System Entity Structure method; (2) discovery of a transition system, using process discovery techniques; (3) integration of fuzzy methods to automatically generate a Fuzzy-DEVS model from the transition system. The last stage is implemented as a plugin in the Process Mining Framework (ProM) environment. A case study is presented in which Fuzzy-DEVS model is inferred from real life data, and the SimStudio tool is used for its simulation.

Keywords: System inference, process mining, fuzzy-DEVS, system entity structure, event logs

1. Introduction

Systems in different areas are becoming more and more complex. Indeed, they consist of multiple and heterogeneous components and intricate interactions among these components. System Theory provides a fundamental framework to understand dynamical systems [1]. In such a framework, a system is characterized by its structure and its behavior. All the knowledge about the system can be organized in a 4-level hierarchy proposed by Klir [2]. This hierarchy is depicted by Fig. 1. It is organized as follows:

- the source level identifies a portion of the real world we are going to observe and measure;
- the data level corresponds to the set of measurements made on the system from its observation;
- the generative level uses formulas or equations to constitute a knowledge;
- the structure level describes the component systems that are interconnected together to form the entire system.

The system structure is defined by the top levels of the hierarchy and the system behavior by the bottom levels. Actually, the more one goes down the hierarchy, the less knowledge is acquired, and conversely. Moving between those levels of system knowledge can be interpreted in three basic ways:

*Corresponding author. Yan Wang, IMS, University of Bordeaux, 33405 Talence Cedex, France. E-mail: yan.wang@u-bordeaux.fr.

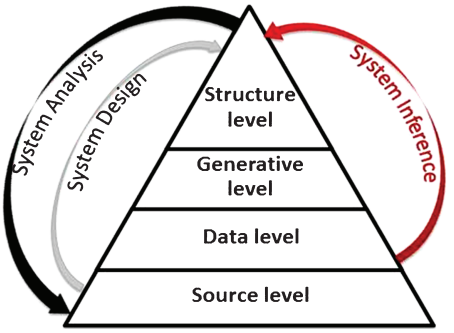

Fig. 1. System problems and hierarchy of knowledge.

– in system analysis, we know the system structure (existing or hypothetical) and we try to generate its data;
– in system design, the system does not exist yet and we are investigating the alternative structures for a completely new system;
– in system inference, the system exists and we are trying to generate its structure from known evidence of its behavior. This has been called "climbing the hill" by Zeigler [3]. Note that a slight but very significant difference between system design and system inference is the system existence or not, prior to the study.

System inference is recognized as a very challenging problem. Process mining [4], as a relatively young area, provides techniques to infer from knowledge given at data level, corresponding knowledge at the structure level. However, process mining requires data be organized in event logs. Event logs are recorded by Extensible Event Stream (XES) standard (in Section 3.3, XES will be explained). Therefore, not every data can be accepted. Moreover, timing aspects are not usually taken into account. Also, there is a lack of modularity, making the design of hierarchical models difficult to realize. Therefore, process mining shows limitations in inferring complex systems.

A new method is proposed to overcome these limitations, which is rooted in the system-theoretic power of the DEVS formalism [3]. DEVS not only has a general framework for Modeling and Simulation (M&S) of complex systems, but also has formal temporal and hierarchical coupling features. It provides a universal way to represent dynamic systems regardless of the application area. Fuzzy-DEVS [5] adds to DEVS capabilities to take frequency of events and imprecise knowledge into account by applying fuzzy sets theory [6]. Another candidate of the formalism for providing similar advantages is Stochastic DEVS model [7], yet Fuzzy-DEVS is more convenient than Stochastic DEVS for the following reasons:

– the concept of possibility used in Fuzzy-DEVS emphasizes the likelihood of an event in the system in a more objective manner than the concept of probability used in Stochastic DEVS;
– the concept of possibility allows users to focus only on the mainstream behavior of the business process;
– Fuzzy-DEVS can provide more semantics by integrating subjective data and linguistics.

This paper proposes a new model-based system inference method based on the construction of a Fuzzy-DEVS model from real event data. This method, called D2FD (Data to Fuzzy-DEVS), makes use of the Two Phase Approach [4] known in process mining. In the earlier stages of D2FD, we propose a structured method to extract event logs from the raw event data using the System Entity Structure (SES) framework. In the final stages, we propose a method to automatically generate a Fuzzy-DEVS model from the event logs using Dependency Method to handle frequency, and Adapted Fuzzy Time Controller AFTC to handle timing aspects. Possibility Measures and the weighted average method are then used, and the resulting model is simulated with the SimStudio package (a Java implementation of DEVS). The simulation results are used to improve the business process behind the data collected.

The paper is organized as follows: related works are presented in Section 2. Background information, i.e. Fuzzy-DEVS, SES, and event logs, are introduced in Section 3. Section 4 presents the D2FD method. Its implementation in the ProM and the use of the simulation tool SimStudio are presented in Section 5. Section 6 shows a case study and Section 7 gives conclusion and perspectives.

2. Related works

It has been shown that process mining can audit relevant information from event logs [8]. A conceptual approach is proposed in [9] to extract event data from databases. Both studies provide general and abstract methods without results from real case study.

In process mining [4], the resulting process model is usually a Petri net model [10]. Two techniques exist to discover the Petri net model: the α-algorithm and the Two Phase Approach. The α-algorithm is able to

discover concurrency but is unable to take frequencies into account. The Two Phase Approach (it will be explained in section 4.2) first transforms an event log into a low-level transition system [11] and then synthesizes a Petri net from the transition system. A major drawback of this approach is that the discovered Petri net cannot represent the timing aspects.

Fuzzy approaches such as fuzzy reasoning, fuzzy modeling and fuzzy inference systems have been used to address the issue of incomplete knowledge in complex systems modeling. Giambiasi et al. [12] propose logic gates with fuzzy delays for modeling and simulation. Zeigler et al. [13] propose to integrate genetic algorithm and fuzzy inference system with DEVS. This genetic algorithm can be extended by a multilevel resolution search strategy in order to solve different degrees of abstracted problems [14]. Dahmani and Hamri [15] use if-then rule to fuzzy controller and specify the duration of state in the DEVS model. Bisgambiglia et al. [16] use fuzzy inference systems (FIS) with DEVS formalism in order to perform the control or the learning on systems described incompletely or with linguistic data. Santucci and Capocchi [17] propose an approach based on the use of Fuzzy Control Language allowing facilitating the modeling and simulation of DEVS. The limitation of these studies is that the model is not constructed from real data.

3. Background

3.1. Fuzzy-DEVS formalism

The Fuzzy-DEVS formalism [5] extends the DEVS formalism with fuzzy set logic. It is able to analysis and describe complex system when the structure of a system is only partially unknown. A fuzzy DEVS model $\tilde{M}$ describes a system as a structure:

$$\tilde{M} = < X,\ Y,\ S,\ \tilde{\delta}_{int},\ \tilde{\delta}_{ext},\ \tilde{\lambda},\ \widetilde{ta} >,$$

where

- X: the set of input values.
- Y: the set of output values.
- S: the set of states.
- $\tilde{\delta}_{int}$: $S \times S \longrightarrow [0, 1]$, fuzzy internal transition function.
- $\tilde{\delta}_{ext}$: $Q \times X \times S \times S \longrightarrow [0, 1]$, fuzzy external transition function, $Q = \{(s, e) \mid s \in S\}$, $0 \leq e \leq ta(s)\}$ where ta(s) is the defuzzified value of $\widetilde{ta}$.
- $\tilde{\lambda} : S \times Y \rightarrow [0, 1]$, fuzzy output function.
- $\widetilde{ta} : S \times \tilde{A} \rightarrow [0, 1]$, fuzzy time advance function, $\tilde{A}$ = the set of fuzzy linguistic numbers.

Here, the DEVS concepts of internal transition, external transition, output function and time advance function, are integrated with fuzzy set logic. Fuzzy internal transition and fuzzy external transition provide state transitions with fuzzy relation which represents the possibility of each state transition. Fuzzy output function also provides the output function with fuzzy relation to represent possibilities of output event. Fuzzy time advance function is extended to a fuzzy set $\tilde{A}$ which represents fuzzy linguistic numbers.

3.2. System entity structure

The System Entity Structure [18] approach defines an ontological framework to represent M&S knowledge in a hierarchical manner. Figure 2 shows the basic elements of the SES. Entities represent things that exist in the real world. They can be assigned with variables, which provide values within given ranges and types. Aspects represent ways of decomposing entities into more detailed parts. Multi-aspects are aspects for which the components are all of one kind. Specializations represent categories in specific forms that an entity can assume.

3.3. Event logs

The event logs are based on the XES standard [4]. An XES document (i.e., XML file) contains one log which is related to one specific process. A log can contain any number of traces. Each trace describes a sequential list of events corresponding to a particular case. The log, its traces, and its events may have any number of attributes. Attributes are defined by the type of data value they represent. An

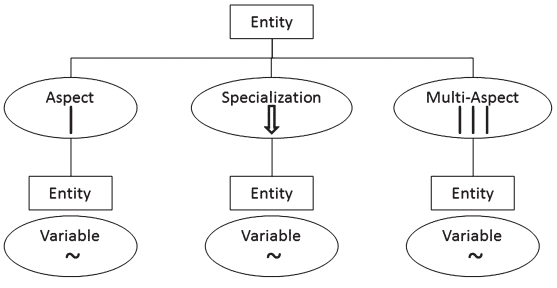

Fig. 2. Basic SES representation.

extension defines a set of attributes for a specific perspective. The concept extension defines an attribute which stores commonly understood names of type elements. Similarly, the time extension stores the time information.

4. D2FD method

The D2FD method we propose to climb the hierarchy of system knowledge, from data to structure, has three major stages, as presented in Fig. 3: (1) from event data to event logs; (2) from event logs to transition system; (3) from transition system to Fuzzy-DEVS model. Here, a toy case of an e-shopping company is used to support the presentation of the D2FD method. Later a real-world case will be shown.

4.1. From event data to event logs

In Fig. 3, the world cloud is related to the source level of Fig. 1. It contains people, machines, organizations and so on. The focus of this paper is on the output side of the world, i.e., event data. Events can be of various types, and they can be recorded in various ways. Besides the twelve guidelines given in [9], we propose four more guidelines for a proper handling of event data:

- event data must be recorded in csv or excel files, and these documents are related to each other;
- activity names should be simple, precise and clear. Similar name can have similar meaning;
- as suggested by Fig. 5, event data correspond to activities that are structured into attributes (i.e., properties that define the event, among which start time and finish time are mandatory), case (each event refers to a case), and instance (a specific sequence of case);
- events are ranked, firstly by instance, and secondly increasingly by start time.

In the e-shopping toy case we consider, there is a start document containing company requests and an end document containing customer orders. In Table 1, the e-shopping products (Pro field) indicate instances (X is the name of the product). One product has several orders (O field), each of which being a case at a different time (the ST field indicates the start time, and the ET field indicates the end time). Attributes are the e-shopping shops (Shop field), the product departments (PD field) and the product subthemes (PS field). In Table 2, the customers (CM field) are the instances (Y is the name of the customer). One customer can place several orders (O field), each being a case at a different time (the ET and ST fields have the same meaning than previously). Attributes are the e-shopping shops (Shop field), and the product departments (PD field).

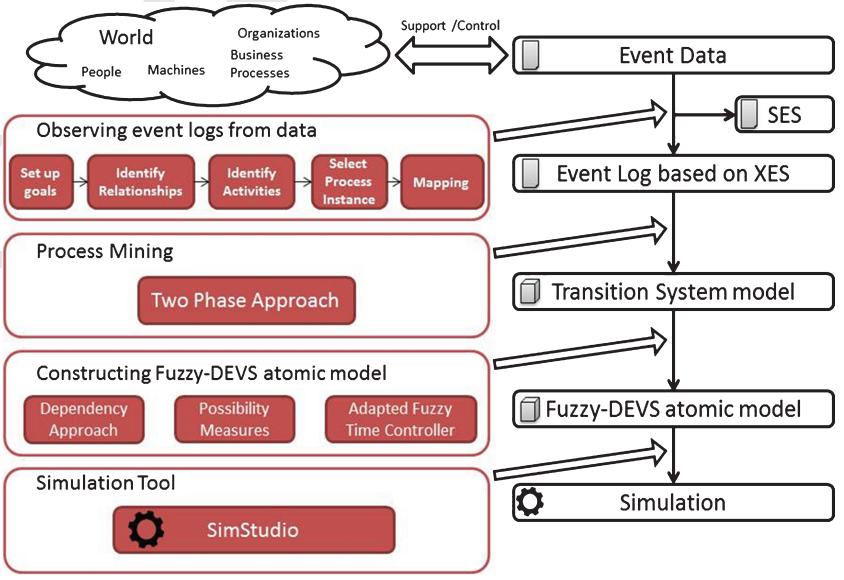

Fig. 3. General structure of the D2FD method.

Table 1
Start document of toy case

Pro	O	ST	ET	Shop	PD	PS
X_1	1	ST_1	ET_1	Peri	Clothing	Women
X_1	2	ST_2	ET_2	Peri	Clothing	Women
X_2	1	ST_3	ET_3	Peri	Clothing	Men
X_3	1	ST_4	ET_4	Peri	Clothing	Luggage
X_4	1	ST_5	ET_5	Peri	Electronic	TV & video
X_5	1	ST_6	ET_6	Peri	Electronic	Computer

Table 2
End document of toy case

CM	O	ST	ET	Shop	PD
Y_1	1	ST_7	ET_7	Peri	Sports
Y_2	1	ST_8	ET_8	Peri	Electronic
Y_2	2	ST_9	ET_9	Peri	Electronic
Y_2	3	ST_{10}	ET_{10}	Peri	Electronic

The stage of D2FD method corresponding to observing event logs from event data is composed of five steps: (1) goals definition, (2) relationships identification, (3) activities identification, (4) process instance selection, and (5) mapping to XES file. These steps are explained hereafter.

4.1.1. Goals definition

Prior to any other activity in the method is the definition of study goals. Goals are investigated from interview. They include the problem to be addressed or the performance to evaluate.

In the toy case, two goals are observed:

- how does the e-shopping company work? and
- how does the customer choose products?

4.1.2. Relationships identification

Raw data are not often well organized, and there is more than one way to build event logs that can store them. Identifying underlying relationships can help researchers find analysis results. SES is used to achieve this identification.

Santucci et al. [19] propose an extension of SES in order to integrate the concepts of abstraction hierarchy into DEVS. Cheon et al. [20] propose a method and an example to generate an experimental frame by using SES from the source data. Although most of researches propose to use SES to construct DEVS models, they do not use SES to discover the mechanisms of internal transition and external transition. To our knowledge, there is no method proposed in the literature to build Fuzzy-DEVS model from real data, using SES.

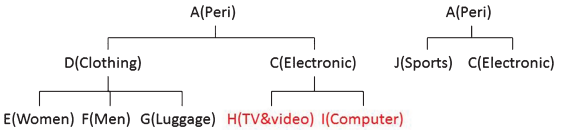

Fig. 4. Two generated SES structure from the toy case.

We propose that each document generates one SES structure as well as one Fuzzy-DEVS atomic model. Each attribute in a document defines a level in the corresponding SES structure.

In the toy case, two SES structures are constructed as shown in Fig. 4. The start document has three attributes, and the end document has two attributes. A, C, D, E, F, G, H, I, and J are different activities, which represent the elements in Tables 1 and 2. In Table 1, D and C are the aspects of A, while E, F, and G are the aspects of D, and H and I are the aspects of C. Respectively, in Table 2, J and C are the aspects of A.

4.1.3. Activities identification

In addition, we propose a rule to find modularity between documents, by distinguishing public and private activity. The rule is shown as follows: (1) if some activities have a strong relationship with the activities in other documents, we identify these activities as public activities and their children activities as private activities; (2) Activities without relationship with any other ones, are identified as public.

In the toy case of Fig. 4, activity C exists in both SES structures. Therefore, the children of C, i.e., H and I, are private activities (written in red). A, C, D, E, F, G, and J are public activities.

4.1.4. Process instance selection

The process instance [21] is the object that one follows throughout the business process. In a document-based business process, the process instance to be selected is related to at least one document. In some cases, all relevant knowledge resides in a single document (e.g., the start document). In other cases, the knowledge is spread among various documents.

Based on the tree structure of SES, there are different levels of activities. The problem exists when the activities can be either public activity or private activity. In Fig. 4, D and C are possible to become private activities as A is similar in both structures. We propose to select one level (or we can say one attribute) which involves the key and interesting activities.

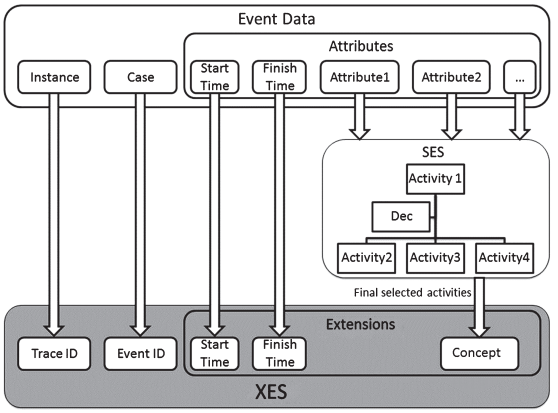

Fig. 5. Mapping between event data, SES and XES.

In the toy case, both the start document and the end document are involved in the process instance selection. The key activities are the leaves of the SES structures build (i.e., {E, F, G, H, I} at one side, and {J, C} at the other side).

4.1.5. Mapping to XES file

In Fig. 5, a mapping from event data to event logs is proposed to convert instance to trace, case to event id, time to time extension, and activities to concept extension.

4.2. From event logs to transition system

The Two Phase Approach in process mining [4] provides a discovery technique, which first transforms an event log into a low-level transition system and then synthesizes a Petri net from transition system. The D2FD approach reuses the first stage of the Two Phase Approach (i.e., the production of a transition system).

In the toy case, the set L of event logs extracted, using traces in the company documents is the following: $L = [(E, F, G, H)^{10}, (E, G, F, H)^{15}, (E, I, H)^{3}]$, where numbers in exponent indicate the times of the corresponding traces. Every position in a trace corresponds to a state in the resulting transition system. For example, when the current state is between F and G, the partial trace $\sigma_{\text{past}} = \{E, F\}$ describes the past of the corresponding case, and the partial trace $\sigma_{future} = \{G, H\}$ describes the future of the corresponding case. The final transition system of the toy case is the upper level of Fig. 6.

4.3. From transition system to fuzzy-DEVS model

A transition system [11] is the process model from which the D2FD method builds a Fuzzy-DEVS model, as depicted by Fig. 3. In a previous work [22],

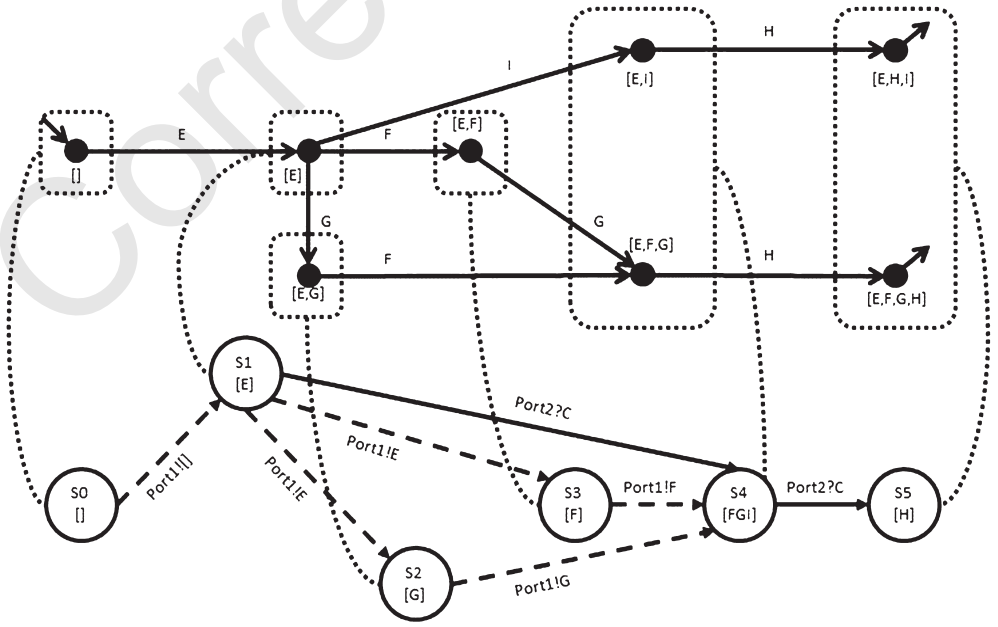

Fig. 6. Toy case from transition system to Fuzzy-DEVS atomic model.

we proposed a region-based approach integrated with some mathematical rules to transform the transition system into functions of the Fuzzy-DEVS model. The four functions in Fuzzy-DEVS (namely, the internal, external, output and time advance functions) are mapped onto fuzzy sets. Dependency Method is used for producing the fuzzy internal transition, fuzzy external transition and fuzzy output function; AFTC is used for obtaining fuzzy time advance function. To execute the Fuzzy-DEVS model, Possibility Measures and the final output of AFTC are applied. The final output of AFTC is inferred by the weighted average method from defuzzification methods.

In this paper, we propose an improved region-based approach. Let $TS = (S^T, A, T)$ be a transition system and $R \subseteq S^T$ be a subset of states. P_a is a period time for each activity $a \in A$. R is a region if for each activity $a \in A$ one of the following conditions holds:

- All transition $(S_1^T,\ a,\ S_2^T) \in T$ enter R, i.e. $s_1^T \notin R$ and $s_2^T \in R$.
- All transition $(s_1^T,\ a,\ s_2^T) \in T$ exit R, i.e. $s_1^T \in R$ and $s_2^T \notin R$.
- All transition $(s_1^T,\ a,\ s_2^T) \in T$ do not cross R, i.e. $s_1^T,\ s_2^T \in R$ or, $s_1^T,\ s_2^T \notin R$.
- For all the transitions $a_1 \in T_1,\ a_2 \in T_2, \ldots, a_n \in T_n$ enter R, $P_{a1} \approx P_{a2} \approx \ldots \approx P_{an}$.

Let pa be the private activity and ua be the public activity. According to Fuzzy-DEVS formalism in section 3.3, the transformation follows the rules:

$$R \to S \tag{1}$$

Where the state of Fuzzy-DEVS atomic model $s \in S$.

$$ua \to x \cup y \tag{2}$$

Where the input value $x \in X$ and the output value $y \in Y$.

$$t\tilde{a} = \begin{cases} 0\ \exists s_0 \in S \\ T^F \\ Infinite\ \neg\exists S = S_1^I \end{cases} \tag{3}$$

Where s_0 is the initial state, T^F is the result coming from AFTC, S_I^I is the input states of all internal transition.

$$T \to \tilde{\delta}_{\text{int}}$$
$$(s_1^T,\ ua,\ s_2^T) \to (s_1,\ s_2,\ \mu_{\text{int}}) \tag{4}$$

Where $s_1 \in R_1$ and $s_2 \in R_2$, μ_{int} is the result coming from Dependency Method in Section 4.2.1.

$$\tilde{\lambda} : (y,\ \mu_{\text{int}}) \tag{5}$$
$$T \to \tilde{\delta}_{ext}$$
$$(s_1^T,\ pa,\ s_2^T) \to (s_1,\ e,\ x,\ s_2,\ \mu_{ext}) \tag{6}$$

Where the elapsed time e: $0 \le e \le \widetilde{ta}$, μ_{ext} is the result coming from Dependency Method.

Figure 6 shows how the transition system (upper level of the figure) of the toy case results in its corresponding Fuzzy-DEVS atomic model (lower level of the figure). As we know, {A, C, D, E, F, G, J} are public activities and {H, I} are private activities. The state sets are split into regions, based on improved region-based approach, and states are converted according to Equation (1). Public activities can be input events or output events, as suggested by Equation (2). Each state contains a unique time life function, as defined by Equation (3). Internal transitions are converted in Equation (4), and output function is defined in on Equation (5). For example, after G expired, S2 transits to S4 and sends event F to port 1. External transitions are converted according to Equation (6). For example, as activity H is private, transition from S1 to S4 is an external transition, triggered by the receipt of C on port 2.

4.3.1. Dependency method with possibility measures

Dependency Method in the heuristic mining [4] is proposed in order to get frequency of events directly from event data. This method helps to generalize internal transition, external transition and output function into fuzzy sets. The first step is to calculate the frequency of every transition from event logs. Then we use Equation (8) to calculate the possibility of every transition.

Possibility Measures [23] can be recognized as one point of view on a fuzzy set. The notion of a fuzzy set corresponds to the need to model an imprecision in knowledge, a gradual transition between close categories, hence the use of membership function. Zadeh [6], Dubois and Prade [23] propose a junction between fuzzy set and possibility theory to represent imprecise possibilities. Possibility Measures is defined as:

$$\forall A,\ \forall B,\ \prod(A \cup B) = \max(\prod(A),\ \prod(B)) \tag{7}$$

The sets A and B in the Possibility Measures can be disjoint. It shows that when concerning about the

disjunctions of the events, we choose the maximum value of the event as the possibility.

In Equation (8), "μ" and "F" respectively defines the possibility and the frequency of a transition from one state to another state respectively. "⟶" means internal or external transition between two states, while "i" and "j" are state numbers.

In the toy case, we can calculate frequencies as follows: $F(s_0 \longrightarrow s_1) = 28$, $F(s_1 \longrightarrow s_2) = 15$, $F(s_1 \longrightarrow s_3) = 10$, $F(s_1 \longrightarrow s_4) = 3$, $F(s_2 \longrightarrow s_4) = 15$, $F(s_3 \longrightarrow s_4) = 10$, $F(s_4 \longrightarrow s_5) = 28$. Based on Equation (8) we can calculate the possibility: $\mu_{int}(s_0 \longrightarrow s_1) = 0.97$, $\mu_{int}(s_1 \longrightarrow s_2) = 0.94$, $\mu_{int}(s_1 \longrightarrow s_3) = 0.91$, $\mu_{ext}(s_1 \longrightarrow s_4) = 0.75$, $\mu_{int}(s_2 \longrightarrow s_4) = 0.94$, $\mu_{int}(s_3 \longrightarrow s_4) = 0.91$, $\mu_{ext}(s_4 \longrightarrow s_5) = 0.97$.

$$\mu(s_i \rightarrow s_j) = \begin{cases} \frac{F(s_i \rightarrow s_j) - F(s_j \rightarrow s_i)}{F(s_i \rightarrow s_j) + F(s_j \rightarrow s_i) + 1} \exists i \neq j \\ \frac{F(s_i \rightarrow s_j)}{F(s_i \rightarrow s_j) + 1} \exists i = j \end{cases} \quad (8)$$

Compared with possibility, probability measures based on the occurrence of events can be defined as:

$$\forall A,\ \forall B,\ A \cap B = \phi,\ P(A \cup B) = P(A) + P(B) \quad (9)$$

The sets A and B in the probability measures are disjoint. It shows that the probability of an event is the percentage of the possibility from all the events, and the sum of the probability of all the events is equal to 1. Compared to probability, possibility is more flexible.

The possibility of the output function is designed to be equal to the possibility of the internal transition. In Fig. 6, state s1 has two internal transitions with the corresponding possibility of 0.94 and 0.91. According to Equation (8), the internal transition $s_1 \longrightarrow s_2$ is selected. For the result of the toy case, when disregarding the external events from the customer model, the process is $s_0 \longrightarrow s_1 \longrightarrow s_2 \longrightarrow s_4$ with the corresponding output function {E, G}. This result evaluates the first goal of the toy case (as expressed in Section 4.1.1). When considering external events from the customer model, the process may be $s_0 \longrightarrow s_1 \longrightarrow s_4$ with no output function. This result evaluates the second goal of toy case.

4.3.2. *Adapted fuzzy time controller*

AFTC is developed from fuzzy discrete event controller system (FDECS) [24]. The structure of AFTC is shown in Fig. 7. From the event logs, every event has a start and an end time. Hence, a multi-set of durations can be derived. The remaining time is calculated from the start of the event to the end of the case. The reason of choosing remaining time as the control to activate the specific fuzzy rule in the AFTC is that the remaining lifetime can be used to get sample measurements for predictions.

The inputs of time duration and remaining time are converted into linguistic variables through fuzzifier. The fuzzifier compares the inputs crisp values with certain levels and generates linguistic values of each input variable for inference kernel connected with knowledge base. The knowledge base includes two parts: membership function, which defines the relations between linguistic variables and time variables; rule base, which characterizes the control of remaining time with a set of linguistic control rules. The inference kernel allows human decision to integrate with fuzzy concepts, membership function and inference rules. The defuzzifier converts the fuzzy sets into crisp value. There are two defuzzifiers in this system:

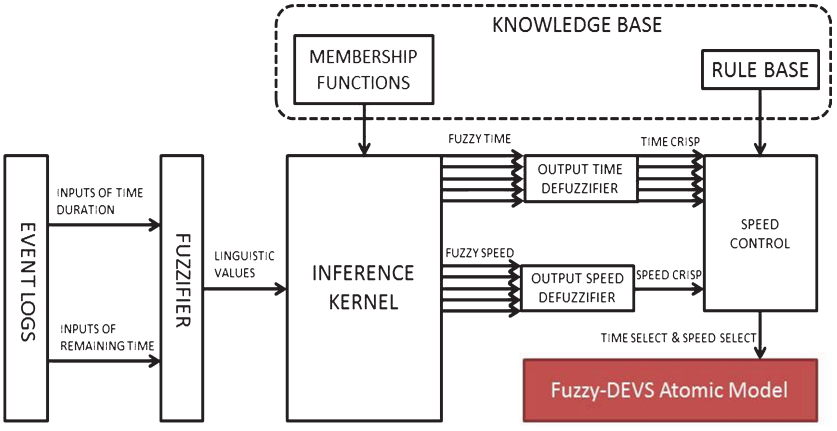

Fig. 7. The structure of AFTC.

time duration and time speed. Five fuzzy time are defuzzified into five crisp time. Five fuzzy speeds are defuzzified into one crisp speed. The defuzzifier is based on the weighted average method as illustrated in Equation (10). Speed control interprets the crisp speed value and activates one of the crisp time values.

$$Z = \frac{\sum \mu(\tilde{z}) \bullet \tilde{z}}{\sum \mu(\tilde{z})} \tag{10}$$

The structure in Fig. 7 is designed for two input variables. Time duration is calculated by subtraction of finish time and start time. Remaining time is calculated by subtraction of finish time of last event and current event. In Table 1 of the toy case, the time duration of X_1 is ET_1 minus ST_1 in the first case. The remaining time of X_1 is ET_2 minus ET_1. We propose the membership functions for the time duration and remaining time as shown in Tables 3 and 4. The possibility is calculated accordingly.

The inference kernel divides the input fuzzy values into two groups. Fuzzy time sets are classified as $\{(\tilde{T}_{VS}, \tilde{T}_S, \tilde{T}_M), (\tilde{T}_{VS}, \tilde{T}_S, \tilde{T}_M, \tilde{T}_B), (\tilde{T}_{VS}, \tilde{T}_S, \tilde{T}_M, \tilde{T}_B, \tilde{T}_{VB}), (\tilde{T}_S, \tilde{T}_M, \tilde{T}_B, \tilde{T}_{VB}), (\tilde{T}_M, \tilde{T}_B, \tilde{T}_{VB})\}$. Fuzzy speed sets include $\{(\tilde{T}_{VL}, \tilde{T}_L, \tilde{T}_A, \tilde{T}_H, \tilde{T}_{VH})\}$. Then fuzzy time sets will be converted into crisp time values $\{T1, T2, T3, T4, T5\}$. Fuzzy speed sets will be converted into crisp speed value $\{S\}$ respectively. The rules to select the crisp time are listed in Table 5. The time T^F can be finally selected as the time of the state in Fuzzy-DEVS model according to the crisp speed value.

Table 3
Membership functions of input fuzzy time duration

Membership Function - MF	Time Duration
Very Small – VS	0–20%
Small – S	20%–40%
Medium – M	40%–60%
Big – B	60%–80%
Very Big –VB	80%–100%

Table 4
Membership functions of input fuzzy remaining time

Membership Function - MF	Remaining time
Very Low – VL	0–20%
Low – L	20%–40%
Adequate – A	40%–60%
High – H	60%–80%
Very High –VH	80%–100%

Table 5
Illustration of rules applied for time selection

No	Range of S	Selection of speed	Selection of time T^F
1	0–20%	Very Fast	T1
2	20%–40%	Fast	T2
3	40%–60%	Medium	T3
4	60%–80%	Slow	T4
5	80%–100%	Very Slow	T5

5. Implementation

ProM is used to implement the D2FD method. ProM is an open-source framework for collecting tools and applications of process mining [4]. Plugins can be defined to extend it.

A new plugin called "Convert to Fuzzy-DEVS using Regions" is designed to extend ProM. This plugin is synchronized on the server of the process mining group [25]. In this plugin, two objects are loaded in the input. One is the event log based on XES standard and the other one is the transition system. The output side is the Fuzzy-DEVS model.

The simulation engine we used to simulate the Fuzzy-DEVS atomic model is SimStudio [26]. The SimStudio environment is based on a DEVS metamodel that offers a Model abstract class, from which derive an Atomic Model abstract class and a Coupled Model abstract class. The models of user must derive from these sub-classes and override the abstract methods. These models are then executed by predefined engines (simulators for atomic models, and coordinators for coupled models, with a root coordinator as the central manager of the simulation time).

We use only atomic models in SimStudio, as the standard DEVS simulation protocol (as implemented in SimStudio) differs from the one of Fuzzy-DEVS when it comes to coupled models.

6. Case study

The case study is conducted on real event data collected from an employee insurance agency in the Northlands [27]. From the description, the agency is interested in insight and recommendations of event data and two main goals can be captured: (1) how the channels are being used; and (2) when are customers moving from one contact channel to the next.

The two interesting documents are "Question.csv" and "Werkmap-message.csv". From "Question.csv", we construct the SES structure shown in Fig. 8. From "Werkmap-message.csv", we build the SES structure shown in Fig. 9. All the words in the figures are

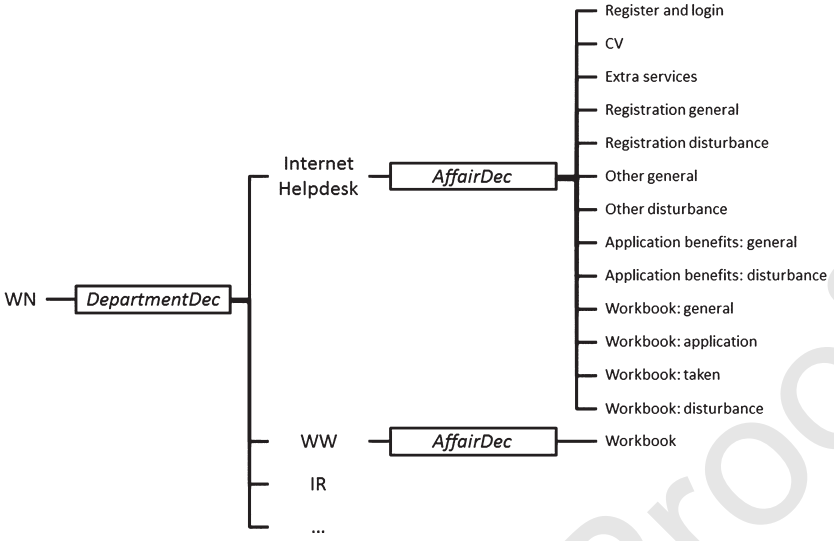

Fig. 8. The SES structure from "Question.csv".

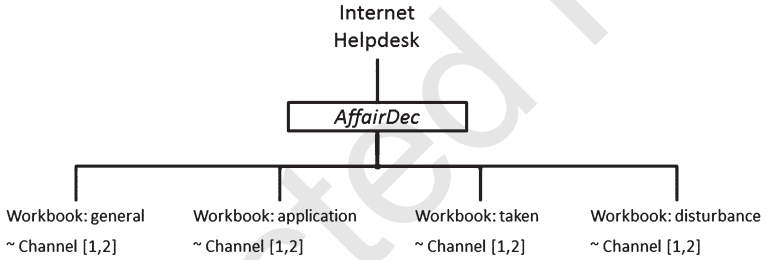

Fig. 9. The SES structure from "Werkmap-message.csv".

translated from Dutch language to English language. In Fig. 8, WN represents the Netherlands agency and it has several departments as aspects. The branch of "Internet Helpdesk" is analyzed as the most interesting part. All the activities of events inside "Internet Helpdesk" are listed as aspects without children activities and they are all public activities. In Fig. 9, all the events are related to "Internet Helpdesk". If we look at four activities ("Workbook: general", "Workbook: application", "Workbook: taken", "Workbook: disturbance") which have similar activities in Fig. 8, their children activities, which are the variables "Channel" between 1 and 2, are identified as private activities.

The next step is to select process instance. "Question.csv" and "Werkmap-message.csv" are selected as start and end document. The interesting level in Fig. 8 is the affair aspect and the interesting level in Fig. 9 relates to channel variables. Moreover, we add "Werkmap" from the department of "WW" and delete "Register and login", "CV", "Extra services" which are useless. So the final selected activities in Fig. 8 are: "Registration general", "Registration disturbance", "Other general", "Other disturbance", "Application benefits: general", "Application benefits: disturbance", "Workbook: general", "Workbook: application", "Workbook: taken", "Workbook: disturbance", "Workbook". The final selected activities in Fig. 9 are: "Channel 1", "Channel 2".

When selecting the final activities, the plugin "Convert CSV to XES" is applied to convert into event logs. The process of mapping is depicted in Fig. 10. "Customer ID" is converted to trace. "ContactTimeStart" is converted to start timestamp and "ContactTimeEnd" is converted to end timestamp. The interesting level is found in the attributes of the file called "QuestionSubtheme" and it is transformed into the name of concept in event logs. At last, we get two XES files, i.e., "Question.xes" and "Werkmap.xes". These two XES files are analyzed one by one through the plugin "Mine Transition System" to construct two corresponding

CSV Preview (1000 rows - scroll down to load more)

Data Type	DISCRETE	...	LITER...	...	...	LITERAL	LITERAL	LITERAL	...	...	...	LITERAL	LITERAL	LITERAL
Data Pattern														
Trace Attribute														
XES Extension Attribute														
Event Attribute	CustomerID	...	Gender	...	...	ContactDate	ContactTimeStart	ContactTimeEnd	...	...	...	QuestionTheme	QuestionSubtheme	QuestionTopic
	CustomerID	...	Gender	...	...	ContactDate	ContactTimeStart	ContactTimeEnd	...	...	...	QuestionTheme	QuestionSubtheme	QuestionTopic
	220	...	V	...	...	30/09/2015	11:55:16.0000000	12:06:26.0000000	...	...	...	WN WB AJD	3. Einde overeenko...	Vaststellingsov...
	220	...	V	...	...	30/09/2015	12:06:27.0000000	12:07:56.0000000	...	...	...	WN WB AJD	3. Einde overeenko...	Opzegtermijn: H...
	220	...	V	...	...	01/10/2015	11:57:15.0000000	12:06:17.0000000	...	...	...	WN WW	Aanvragen (WWZ 1-...	Wat gebeurt er ...
	220	...	V	...	...	15/10/2015	15:15:31.0000000	15:23:56.0000000	...	...	...	WN WB AJD	Contactgegevens AJ...	Specifieke vraag
	220	...	V	...	...	17/11/2015	11:52:11.0000000	11:56:52.0000000	...	...	...	WN WB AJD	4. Ontslagprocedure	Status: Wat is d...
	220	...	V	...	...	16/12/2015	14:42:42.0000000	14:47:05.0000000	...	...	...	WN WW	Formulier Inkomsten...	Algemeen: Wan...
	318	...	M	...	...	15/12/2015	12:24:42.0000000	12:29:44.0000000	...	...	...	WN WW	Correspondentie	Kunt u mij een k...
	318	...	M	...	...	11/02/2016	08:57:41.0000000	09:14:38.0000000	...	...	...	WN WW	Betaling (WWZ 1-7-2...	Wanneer is/wor...
	495	...	M	...	...	21/07/2015	09:49:32.0000000	09:54:26.0000000	...	...	...	WN WW	Recht	Recht: Hoe word...

Fig. 10. Mapping between CSV file and XES file.

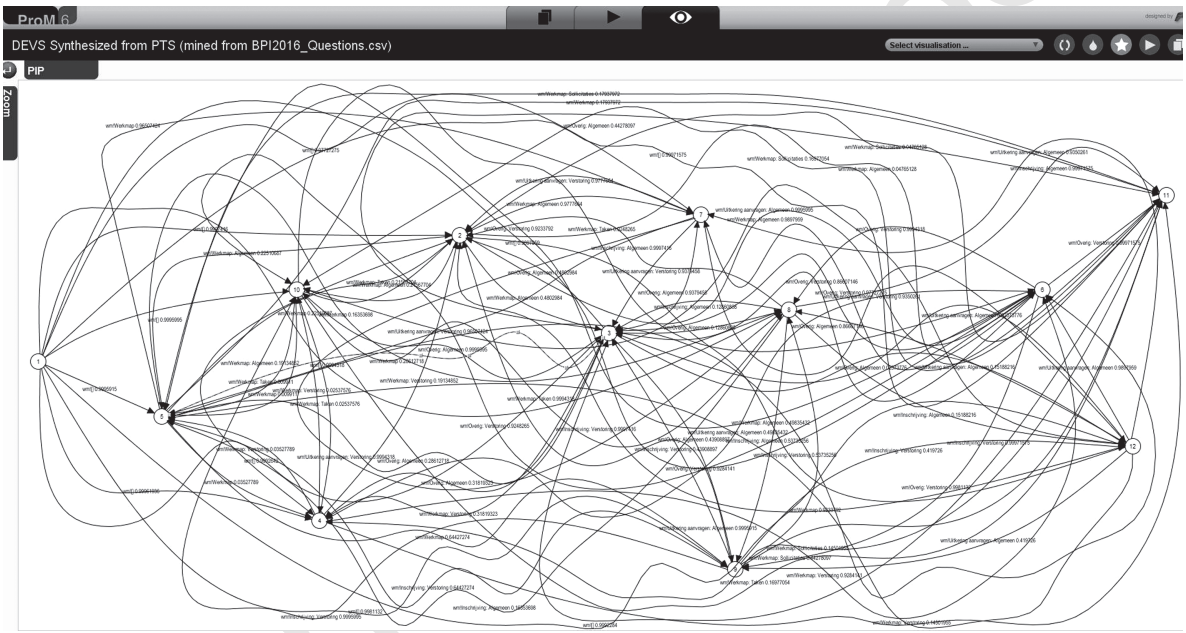

Fig. 11. Fuzzy-DEVS atomic model generated from Question file.

transition system models. Then through the designed plugin "Convert to Fuzzy-DEVS using Regions", two corresponding Fuzzy-DEVS atomic model are generated.

Figure 11 presents the first Fuzzy-DEVS atomic model generated from "Question.xes". Figure 12 shows the corresponding scheme, according to simulation results. Similarly, the second Fuzzy-DEVS atomic model is generated from "Werkmap.xes" and presented in Fig. 13. The first model is like a generator which is consistently sending outputs. The second model is like a processor which is waiting for events. As a result, we consider having one port, called "wm", to connect the two models.

In Fig. 12, as all the activities are the public activities in SES structure, all the transitions are converted into the internal transitions defined as the combination of "wm", "!" and the output event, represented as classical arrow. Behind the graphical notation, the possibility is added. The state represented by the circle. Every state has a unique identity number. The initial state is converted into the state with label of 1. The output event of the initial state is "[]". The business process starts from the initial state and continues the transitions automatically until it reaches the desired state.

In Fig. 13, all the transitions are converted into the external transitions defined as the combination

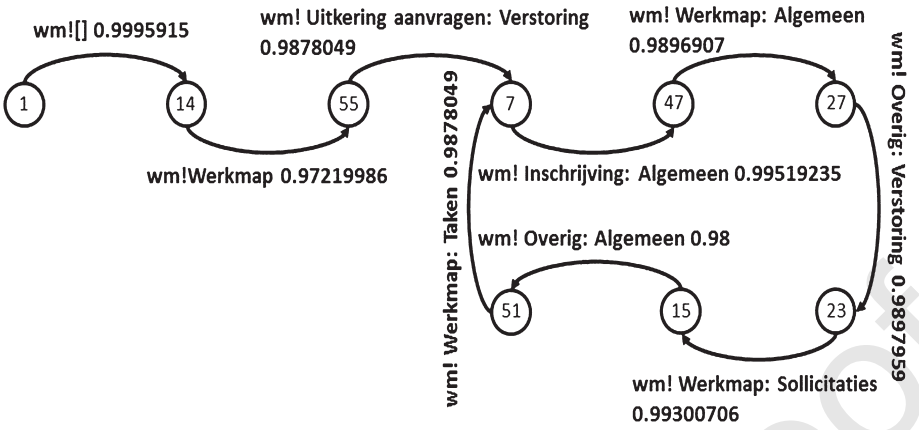

Fig. 12. Represented scheme from Fig. 11.

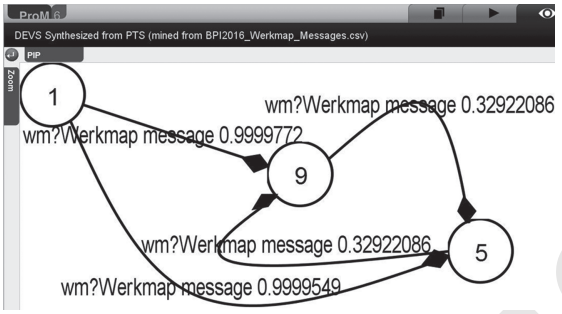

Fig. 13. Fuzzy-DEVS atomic model generated from Werkmap file.

of "wm", "?" and input event, represented as diamond arrow. The initial state is labeled as 1. Every external transition is related to a possibility at the end of the graphical notation. As all the activities are private activities, all the transition are external transition and all the states are set by the infinite time. Here we only get "Werkmap message" (Workbook message in English) instead of four public activities. It is extracted from the parents of variables in "Werkmap.xes". A major issue is the incompleteness of information in "Werkmap.xes". The participation of business people is required to get more precise information.

Each state of Fig. 11 with a leaving internal transition is given a time life function. This time life function is calculated by AFTC. Figure 14 presents one part of the results of the time duration. The initial state is set as the time of 0. This state will immediately go to the next state. In the second state "[Uitkering aanvragen: Verstoring]", time duration and remaining time are identified as inputs. According to the membership function in Tables 3 and 4, we get the possibility of VS, S, M, B, VB, VL, L, A, H, VL. We use the weighted average method in Equation (10) to get the speed "very fast" from remaining time. Then we put this speed in the illustration of rules in Table 5. The final time T1 = 1122.3 seconds. The same calculation happens to the following states.

The SimStudio simulation results are shown in Fig. 15. The numbers displayed relate to the time (converted here into minutes). Every state has several internal transitions. The execution of these internal transition is based Possibility Measures. The initial

```
fuzzy time controller starts
 Time of [[],[]] is 0.0 seconds
fuzzy time controller starts
VS: 0.88 S: 0.0 M: 0.03 B: 0.03 VB: 0.03
VL: 0.82 L: 0.06 A: 0.03 H: 0.03 VH: 0.03
fuzzyspeed: 3234.4312499999996
Selection of speed: very fast
 Time of [[Uitkering aanvragen: Verstoring],[Uitkering aanvragen: Verstoring+]] is 1122.3 seconds
fuzzy time controller starts
VS: 0.99 S: 0.01 M: 0.0 B: 0.0 VB: 0.0
VL: 0.29 L: 0.71 A: 0.0 H: 0.0 VH: 0.0
fuzzyspeed: 3301.161844116149
Selection of speed: fast
 Time of [[Werkmap],[Werkmap+]] is 2397.69 seconds
```

Fig. 14. Part of results from AFTC.

```
1 : []
Internal transition: [] & [Werkmap: Taken] ---- 0.98
Internal transition: [] & [Werkmap: Verstoring] ---- 0.9944444
Internal transition: [] & [Werkmap: Algemeen] ---- 0.99519235
Internal transition: [] & [Inschrijving: Algemeen] ---- 0.9878049
Internal transition: [] & [Uitkering aanvragen: Verstoring] ---- 0.9714286
Internal transition: [] & [Werkmap] ---- 0.9995915
Internal transition: [] & [Overig: Verstoring] ---- 0.9896907
Internal transition: [] & [Inschrijving: Verstoring] ---- 0.98507464
Internal transition: [] & [Overig: Algemeen] ---- 0.99300706
Internal transition: [] & [Uitkering aanvragen: Algemeen] ---- 0.9861111
Internal transition: [] & [Werkmap: Sollicitaties] ---- 0.9897959
40 : [Werkmap]
Internal transition: [Werkmap] & [Werkmap: Algemeen] ---- 0.8436912
Internal transition: [Werkmap] & [Inschrijving: Algemeen] ---- 0.9355477
Internal transition: [Werkmap] & [Werkmap: Taken] ---- 0.9603524
Internal transition: [Werkmap] & [Werkmap: Sollicitaties] ---- 0.9233792
Internal transition: [Werkmap] & [Inschrijving: Verstoring] ---- 0.94709635
Internal transition: [Werkmap] & [Overig: Algemeen] ---- 0.88996136
Internal transition: [Werkmap] & [Overig: Verstoring] ---- 0.9241352
Internal transition: [Werkmap] & [Uitkering aanvragen: Verstoring] ---- 0.97219986
Internal transition: [Werkmap] & [Werkmap: Verstoring] ---- 0.8633422
58 : [Uitkering aanvragen: Verstoring]
Internal transition: [Uitkering aanvragen: Verstoring] & [Werkmap: Algemeen] ---- 0.71487606
Internal transition: [Uitkering aanvragen: Verstoring] & [Uitkering aanvragen: Algemeen] ---- 0.3490566
Internal transition: [Uitkering aanvragen: Verstoring] & [Werkmap] ---- 0.97219986
Internal transition: [Uitkering aanvragen: Verstoring] & [Overig: Algemeen] ---- 0.6101695
Internal transition: [Uitkering aanvragen: Verstoring] & [Inschrijving: Algemeen] ---- 0.9878049
```

Fig. 15. Part of simulation results from Question document by SimStudio.

state is set as 1 minute. The output event "[]" is sent to the port "wm" as the time series of 1. Then the internal transition "[] & [Werkmap]" is triggered with the maximum possibility 0.9995915. According to the final time of the state, which is 2397.69 seconds (i.e., almost 39 minutes), the output event "[Werkmap]" is sent to the port "wm" at time 40. Later, the internal transition "[Werkmap] & [Uitkering aanvragen: Verstoring]" is triggered. The output event "[Uitkering aanvragen: Verstoring]" is sent to the port "wm" at the time series of 58.

7. Conclusion

This paper presents the D2FD method, an integrated approach to the discovery of discrete event simulation model from real data. This method provides a solution for system inference in System Theory and enhances process discovery in process mining. TheSES framework is used to build a modular and hierarchical abstraction from the data collected, in order to structure the event log that will be extracted. Process mining is then used to generate a transition system, and an improved region-based approach is used to generate a Fuzzy-DEVS model from this transition system. The Dependency Method is used to extract the frequency of events from event logs, and the AFTC is used to extract the timing aspects. These methods expand rough approximations into fuzzy environment and solve issue of imprecise. The D2FD method is implemented on ProM which is practical, visible, automatic and available [25]. The resulting model is simulated by using SimStudio. The D2FD method makes it possible to extract simulation knowledge from the traces of a complex system, and to reveal an optimal business process.

However, this method still needs improvements. Two perspectives to this work are envisioned. First, we anticipate constructing structured and meaningful (therefore coupled) model for business simulation. Coupling mechanisms are needed. Their corresponding simulation protocol can be implemented in the simulation environment. The automated discovery of Fuzzy-DEVS coupled models means an enrichment of the D2FD method, so that the coupling knowledge is carried in the ontology. Secondly, the validation of the discovered model remains a challenge. It needs the participation of domain experts. For example, interviews held between process mining users and companies can help to check whether part of the process model or business process need to be improved.

References

[1] H.A. Simon, The Architecture of Complexity, *Facets of Systems Science*, Springer US, 1991, pp. 457–476.

[2] G. Klir, *Architecture of Systems Problem Solving*, Springer Science & Business Media, 2013.
[3] B.P. Zeigler, H. Praehofer and T.G. Kim, *Theory of Modeling and Simulation*, Academic Press, New York, 2000.
[4] W.M.P. Van der Aalst, Process Mining: Discovery, *Conformance and Enhancement of Business Processes*, Springer Science & Business Media, 2011.
[5] Y.W. Kwon, H.C. Park, S. Jung and T.G. Kim, Fuzzy-DEVS Formalisme: Concepts, Realization and Application, *Proceedings AIS 1996*, 1996, pp. 227–234.
[6] L.A. Zadeh, Fuzzy sets, *Information and Control* **8** (1965), 338–353.
[7] R. Castro, E. Kofman and G. Wainer, A Formal Framework for Stochastic DEVS Modeling and Simulation, *Proceedings of the Spring Simulation Multiconference, Society for Computer Simulation International*, 2008, pp. 421–428.
[8] M. Jans, M.G. Alles and M.A. Vasarhelyi, A field study on the use of process mining of event logs as an analytical procedure in auditing, *The Accounting Review* **89**(5) (2014), 1751–1773.
[9] W.M.P. Van der Aalst, Extracting Event Data from Databases to Unleash Process Mining, *BPM-Driving Innovation in a Digital World*, Springer International Publishing, 2015, pp. 105–128.
[10] J.L. Peterson, Petri nets, *ACM Computing Surveys* **9**(3) (1977), 223–252.
[11] M.K. Robert, Formal verification of parallel programs, *Communications of the ACM* **19**(7) (1976), 371–384.
[12] N. Giambiasi, M. Smaili and C. Frydman, Discrete Event Simulation with Fuzzy Times. *In European Simulation Symposium*, Turkey, 1994.
[13] B.P. Zeigler, Y. Moon, V.L. Lopes and J. Kim, DEVS approximation of infiltration using genetic algorithm optimization of a fuzzy system, *Mathematical and Computer Modeling* **23**(11-12) (1996), 215–228.
[14] J. Kim and B.P. Zeigler, Designing fuzzy logic controllers using a multiresolutional search paradigm, *IEEE Transactions on Fuzzy Systems* **4**(3) (1996), 213–226.
[15] Y. Dahmani and M.E.A. Hamri, *Specification of the State Lifetime in the DEVS Formalism by Fuzzy Controller*, arXiv preprint arXiv:1401.5638, 2014.
[16] P.A. Bisgambiglia, L. Capocchi, P. Bisgambiglia and S. Garredu, Fuzzy Inference Models for Discrete Event systems, *Fuzzy Systems (FUZZ), IEEE International Conference on IEEE*, 2010, pp. 1–8.
[17] J.F. Santucci and L. Capocchi, Fuzzy Discrete-Event Systems Modeling and Simulation with Fuzzy Control Language and DEVS Formalism, *Sixth International Conference on Advances in System Simulation*, 2014, pp. 250–255.
[18] B.P. Zeigler and P.E. Hammonds, *Modeling & Simulation-Based Data Engineering: Introducing Pragmatics into Ontologies for Net-Centric Information Exchange*, Academic Press, 2007.
[19] J.F. Santucci, L. Capocchi and B.P. Zeigler, SES Extension to Integrate Abstraction Hierarchy into DEVS Modeling and Simulation, *Proceedings of the Symposium on Theory of Modeling & Simulation: DEVS Integrative M&S Symposium Society for Computer Simulation International*, 2015, pp. 17–24.
[20] S. Cheon and B.P. Zeigler, Experimental frame Structuring and Aggregation of Source Data: Application to us Climate Normal, *Proceedings of the Spring Simulation Multiconference* Volume 2, 2007, pp. 243–248.
[21] M. Jans, *From Data to Event Log*, Process Mining Camp, 2015.
[22] Y. Wang, G. Zacharewicz, D. Chen and M.K. Traoré, A Proposal of Using DEVS Model for Process Mining, *Proceedings of the European Modeling and Simulation Symposium*, Bergeggi, 2015, pp. 403–409.
[23] D. Dubois and H. Prade, *Possibility Theory: An Approach to Computerized Processing of Uncertainty*, Springer Science & Business Media, 2012.
[24] M.S. Khan, Fuzzy Time Control Modeling of Discrete Event Systems, *International Conference on Intelligent Automation and Robotics*, WCECS, USA, 2008, pp. 683–688.
[25] Y. Wang, The Subversion Server at the Technical University of Eindhoven, 2016. https://svn.win.tue.nl/repos/prom/Packages/TS2DEVS
[26] M.K. Traoré, Simstudio: A Next Generation Modeling and Simulation Framework, *Proceedings of the 1st International Conference on Simulation Tools and Techniques for Communications*, Networks and Systems & Workshops, ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2008.
[27] 12th International Workshop on Business Process Intelligence (BPI) Challenge, 2016. https://data.4tu.nl/repository/collection:event_logs_real