Aggregating Individual Models of Decision-Making Processes

Razvan Petrusel

Faculty of Economics and Business Administration, Babes-Bolyai University,
Teodor Mihali str. 58-60, 400591 Cluj-Napoca, Romania
razvan.petrusel@econ.ubbcluj.ro

Abstract. When faced with a difficult decision, it would be nice to have access to a model that shows the essence of what others did in the same situation. Such a model should show what, and in which sequence, needs to be done so that alternatives can be correctly determined and criterions can be carefully considered. To make it trustworthy, the model should be mined from a large number of previous instances of similar decisions. Our decision-process mining framework aims to capture, in logs, the processes of large numbers of individuals and extract meaningful models from those logs. This paper shows how individual decision data models can be aggregated into a single model and how less frequent behavior can be removed from the aggregated model. We also argue that main process mining algorithms perform poorly on decision logs.

Keywords: decision data model, decision process mining, product based workflow design.

1 Introduction

Using the collective intelligence (e.g. opinions, preferences expressed by many individuals) in various ways is a hot topic both for research and practice. The basic idea behind a lot of approaches in this area is that 'if a lot of people prefer, recommend or simply do something, it must be valuable'. This is the approach which works very well in various contexts (e.g. Google, Wikipedia, Amazon, etc.). But what if we apply this simple, yet powerful, principle in decision making? For example, what if somebody wants to buy a car, or a laptop, or to make an investment? There are a lot of different decision alternatives to choose from and there are even a larger number of criterions to be considered. So, wouldn't it be nice to have, as a guide, some sort of a model that depicts the essence of what a large number of other individuals did, when faced with the same decision? Of course, there are a lot of recommender systems that do just that. Fundamentally, our approach is different. We aim to extract a workflow model that depicts the sequence of actions that were performed in order to choose a decision alternative. We do not aim to indicate which alternative should be chosen, but to show the user and lead him through the 'right' path of actions, so that a fully informed decision is made. 'Right' is the essence of

J. Ralyté et al. (Eds.): CAiSE 2012, LNCS 7328, pp. 47–63, 2012.

what many others have done in a similar situation. For example, we can look at the decision problem of buying shares on the stock exchange. The model can indicate that the decision makers first need to check the quotations of some stock, then compare it with some other stocks, then calculate some indicators, then check the upcoming changes of laws, and so on. From such a model can benefit decision makers with different background and knowledge regarding the issue at hand. For example, individuals who never used the stock exchange can find out what to do and when to do it; while experts can compare their approach against what others have done. It is outside our goal to recommend which choice to make (e.g. that x shares from Y company should be purchased).

The goal introduced above is quite ambitious, and raises a lot of problems. Therefore, some limitation and formalization is needed. We define the overall goal of our research as: 'create a model of the decision making process by extracting it from a large number of individuals'. But, the decision making process is unstructured and is performed (mostly) mentally. So, we need to focus on two sub-problems. The first one is the problem of extracting the flow of mental activities from decision makers and, the second one, finding the right approach to convert what was extracted into a meaningful model. This paper focuses on providing answers to the second sub-problem, while the first one was already addressed in [1].

Our approach relies on using any software that provides data needed for a decision and is enhanced with the feature of logging what the user does while making the decision. Then, we can mine the log and create a model. Extracting process models from logged data is the basic approach in process mining. Therefore, our effort can be placed in this research area. We are interested, at this point, in the data perspective of the decision making process (i.e. we are focusing on the data items used by the decision maker and on the sequence of data derivations).

The paper is organized as follows. In the next section we provide the formal approach for the aggregation of decision making models. In the third section we give an overview of the framework we propose. In the fourth section we aim to validate our approach by showing a running example, a case study, and an experiment. In the last two sections we go through the related work and the conclusions.

2 The Formal Approach

In this section we define the notions that are essential for the aggregation of decision making processes.

***Definition 1 (Data Element)*:** Let D be a set of data elements d, N be a set of data element names n and V be a set of data elements values v. By definition, $d \in D$ is a pair $d = (n, v)$. As shorthand we use $d.n$ to refer to the name of a data element and $d.v$ to refer to its value. For example, d = (income, 2000) $\in$ D is a data element where $d.n$ = "income" and $d.v$ = 2000.

Property 1 (Value of a Data Element): The value of a data element is assigned when that element is created and it doesn't change over time. Given $d_i, d_j \in D$ if $d_i.n = d_j.n \Longrightarrow d_i.v = d_j.v$

***Definition 2 (Basic Data Element)*:** A basic data element, $bd \in BD$ where $BD \subset D$ is a piece of data readily available, which can be used without any additional processing. In the case of decision making, this is data known/found out explicitly by the decision maker. In the case of software a *bd* is data that is displayed to the user.

***Definition 3 (Inputted Data Element)*:** An inputted data element, $id \in ID$ where $ID \subset D$ is a piece of data which is not available explicitly but is known (e.g. the number of months in a year may not be explicitly available). In the case of decision making, this is data that the decision maker knows implicitly.

***Definition 4 (Derived Data Element)*:** A derived data element, $dd \in DD$ where $DD \subset D$ is a piece of data that is created by the decision maker in the decision making process. The logic behind such a data element is that there are some input data elements, $i_1, i_n \in D$, which, through some derivation, are converted to an output element $o_1 \in DD$. The inputs are data elements (*bd*, *id* or *dd*) that must be available at the time the derivation takes place. This needs to be seen as a cause-effect situation. All causes must be enabled before the effect is produced.

Property 3 (Data Derivation): If a derivation is observable, the inputs and the outputs will always be visible, even if the derivation itself is a black-box. For example, in stock exchange the share quotations are the input data which, through some derivations, is converted to the Stock Exchange Index (e.g. Dow Jones). For decision-aware software, if a mental activity can be observed, the inputs and outputs should be visible in the logs.

***Definition 5 (Decision Activity)*:** The notion of decision activity is used to define the conscious mental action of a decision maker to transform some inputs into some outputs. Since we look only at data elements at this time, a decision activity is a data transformation (e.g. using mathematical operations) of some input data elements (*bd*, *id* or *dd*) into some derived data elements *dd*.

In process mining, a basic trace is a finite sequence (σ) of activities (a) denoted as $\sigma = (a_1, a_2, \ldots, a_n)$ [2]. Such a trace encodes the sequence of activities but it does not show any explicit dependency between the activities. In process mining [2], it is assumed that, given the sequence $a_i, a_{i+1} \Longrightarrow a_i$ is the cause of a_j because it is a predecessor in the observed behavior. Since the decision making process is less structured than a business process, such an assumption does not hold (i.e. a decision maker might perform a set of activities in a random sequence). Therefore, we need to define a new trace format which formalizes the dependency in an explicit manner.

***Definition 6 (Decision Trace)*:** A decision log (DL) is defined as a multiset of decision traces. A decision trace (DT) is a tuple (D, DPF, T) where:

- D the set of data elements *d*, $D = BD \cup ID \cup DD$:
- DPF is a set of dpf, where dpf is a function that describes the dependency of each DD as dpf: $D \rightarrow DD$;
- T is a set of t, where t is a timestamp.

Since a DT is defined as a tuple, a sequence of data elements is implied. Note that a derived data element depends on other data elements (i.e. is a consequence of other

data elements). Therefore, in the tuple, the 'source' data elements will show up before the 'consequence' derived data element. Basic or inputted data elements need no transformation; therefore the input element is the empty set.

To give the reader a better understanding of the definition introduced above, we will show a running example of three partial traces from the case study in the paper.

- DT1 = ⟨((income, 2000), ø, time1), ((savings, 50000), ø, time2), ((property_price, 100000), ø, time3), ((dd1, -50000), (savings, property_price), time4), ((period, 240), ø, time5), ((dd2, 208.33), (dd1, period), time6), ((dd3, 1791.67), (income, dd2), time7), ((dd4, 25), (dd1, income), time8)⟩
- DT2 = ⟨((property_price, 100000), ø, time9), ((savings, 50000), ø, time10), ((period, 240), ø, time11), ((dd1, -50000), (savings, property_price), time12), ((dd2, 208.33), (dd1, period), time13), ((income, 2000), ø, time14), ((dd3, 25), (dd1, income), time15), ((dd4, 1791.67), (income, dd2), time16)⟩
- DT3 = ⟨((savings, 50000), ø, time17), ((property_price, 100000), ø, time18), ((dd1, -50000), (savings, property_price), time19), ((income, 2000), ø, time20), ((period, 240), ø, time21), ((dd2, 1791.67), (income, dd1, period), time22), ((dd3, 25), (dd1, income), time23)⟩

Definition 7 (Equivalence of Basic/Inputted Data Elements): Let bd_i and bd_j be two elements of two Decision Traces DT_i and DT_j. $bd_i = bd_j$ if and only if $bd_i.n = bd_j.n$ and $bd_i.v = bd_j.v$ (i.e. both data items have the same names and the same absolute values).

Definition 8 (Equivalence of derived data elements): Let dd_i and dd_j be two elements of two Decision traces DT_i and DT_j. $dd_i = dd_j$ if and only if $(DPF_i, V_i) = (DPF_j, V_j)$, i.e. both data items depend on the same data elements and their absolute values are equal. The basic equivalence looks only at direct predecessors (e.g. for the example decision traces DT2.dd1 = DT3.dd1 but DT2.dd4 ≠ DT3.dd2 because, even if the values are equal, the direct inputs are different). The extended equivalence looks at all the predecessors down to the basic/inputted data items (e.g. in this case DT2.dd4 = DT3.dd2 because DT2.dd4.dpf = {savings, property_price, period, income} and DT3.dd2.dpf = {income, property_price, savings, period} and their value is equal).

Definition 9 (Decision Data Model): A Decision Data Model (DDM) is a tuple (D, O) with:

– D: the set of data elements d, D = BD ∪ ID ∪ DD
– O: the set of operations on the data elements. Each operation, o is a tuple (d, v, DS, t), where:
  - $d \in$ DD, d is the name of the output element of the operation;
  - v is the value outputted by the operation. Can be numeric or Boolean;
  - DS is a list of ds, ds = (ao X d), where:
    - $ao \in$ AO, AO = {+, -, *, /, <>} is a set of arithmetic operations specifying how to produce the output element d based on the input elements ie;
    - $d \in$ D, d is an input data element.
  - $t \in$ T is the set of timestamps at which an operation from O occurs (i.e. the time when the element d is created using o).
– D and O form a hyper-graph H = (D, O), connected and acyclic.

We use as shorthand the notation $o_i.ds$ to refer to the set of input elements for operation o_i and $d.v$ as the value of the data element produced by the operation.

The graphical representation of the DDM created based on DT3, is shown in Fig. 1. The model is annotated with details regarding the operations. We argue that the DDM is easy to understand even without previous knowledge of its semantics. If we look at the model in Fig. 1., it depicts clearly, for example, that the savings must be aggregated with the property price. If we look at the detail of operation 1 we can see that the savings must, actually, be deducted from the property price. The time of the operations is there to provide some sense of sequence, even if it is not strictly enforced by the semantics of the model. We argue that this loose approach is best fitted for decision processes. It balances between the strict process imposed by a workflow model and the free approach of declarative or rule-based models. The DDMs for all the decision traces introduced earlier in this sub-section are shown in Fig. 4.

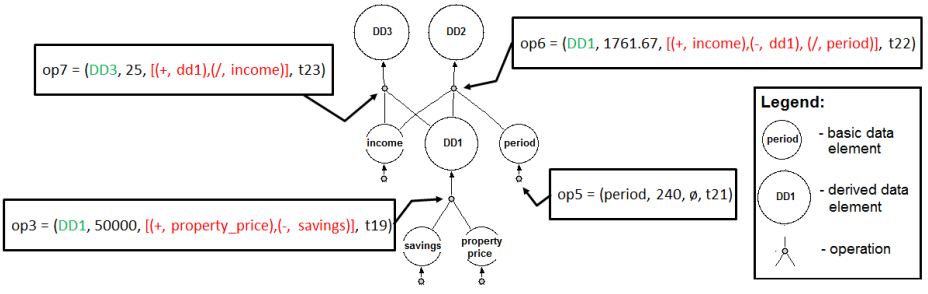

Fig. 1. DDM model of DT3, annotated with the details of the operations

***Definition 10*:** An aggregated DDM is a DDM with the following annotations:

- Frequency of data elements. It indicates how many times a data element is observed in the log;
- Frequency of an operation. It too indicates how many times an operation shows up in a log. There are frequent cases in which the same data element is produced using alternative operations (e.g. the same derived data is created by a single operation with many inputs or by a structured sequence of operations). In Fig. 4 we provide such an example.

Algorithm 1 (creating an aggregated DDM from n DDMs):

Let DDM_1,..., DDM_n be n decision data models. DDMagg is created as follows:

a) Dagg = $D_1 \cup \ldots \cup D_n$
b) Discard duplicates in Dagg
c) Oagg = $O_1 \cup \ldots \cup O_n$
d) Do Find_Equivalent_operation()
   - evaluate every pair (o_i, o_j) where o_i, $o_j \in$ Oagg
   - if $o_i = o_j$ (by extending Definition 8 to DDM operations $o_i = o_j$ if and only if $o_i.ds = o_j.ds$ and $o_i.v = o_j.v$. The user may choose if the basic or extended definition is applied)
     - increase frequency.(o_i) by 1
     - discard o_j

A running example for this algorithm is introduced in sub-section 4.1.

Algorithm 2 (abstracting from an aggregated DDM): The goal of such an algorithm is to provide a 'zoom-in/zoom-out' feature for an aggregated DDM. Zooming-in only shows very frequent behavior while zooming-out also shows less frequent behavior. Let *t* be the frequency threshold so that the abstracted DDM will show only data elements and operations which are observed more frequently than the threshold. The abstracted DDM is created by:

a) Do Extended_Find_Equivalent_operation()
   - evaluate (o_i, o_j) where o_i, $o_j \in$ Oagg
   - if $o_i = o_j$ (using Definition 8 Extended)
     - increase frequency.(o_i) by 1
     - discard o_j

b) If frequency.*oi* < t
   - Remove *oi*

c) If frequency.*di* < t
   - Remove *di*

d) Do Find_unconnected_derived_data
   - If found() create artificial *oj* with only basic or inputted data elements as inputs and *ddj* as output.

The last step is needed because there might be derived data items above the threshold, but produced by several alternative operations with a frequency below the threshold. For example, the threshold may be set to 3 and a derived data element may show up 4 times. But it might be produced by 2 different operations, each with a frequency of 2. So the operations would be removed while the data element would still be in the model. In this case, the artificial operation would link the derived data element directly to the basic data items involved in the derivation. Any intermediary operations below the threshold would be removed. This problem shows only when setting low thresholds (e.g. 2, 3 or 4). For an example please refer to section 4.1.

In order to validate that there is a good match between the recorded events and the DDM, we can use the notion of conformance checking. It is trivial to check the fitness for an instance DDM. Checking it for an aggregated DDM is an issue that will be approached in a separate paper. As a brief preview, we argue that, in a DDM an operation can be looked at as a transition. For replaying the sequence of the actions in the log, we require that all input elements are available when an operation is executed. However, there are differences from a Petri Net due to semantics (i.e. a transition consumes tokens from the input places) which cannot be applied to DDMs. If we strictly apply the fitness measures to an aggregated DDM it is under-fitting (i.e. there are a lot of 'tokens' left behind). But it is not the best validation measure.

3 The Framework of Decision-Making Process Mining

If we move past the main idea stated in the introduction to a more practical approach, we can see that there are two main issues that need to be solved. The first one is to define a knowledge extraction method for the decision making process. This is not a

trivial matter mostly because of: the fuzzy mental processes; the heterogeneity of decision making styles; and the number of individuals that need to be involved in the studies. We proposed the use of simulation software that logs the actions of the users [1]. Such a log needs to clearly capture what are the elements used by the decision maker in the decision making process and the correlation between them.

While keeping in mind that the goal of this research is to automatically extract a model of the data perspective of the decision-making process, as performed by multiple decision makers, we show in Fig. 2, the overview of our approach.

Everything starts with a large number of decision makers that interact with software in order to make some decision. It can be any software, even a simple spreadsheet, which simply provides the users with a set of values that are relevant to the decision to be made. There should also be the possibility to log the values that are used by the decision maker in order to reach the decision. Since we are interested in business decision making, we can use as examples of such software the on-line stock trading platforms, management simulation games, on-line shops, etc. Any software can be upgraded to do the logging by various means (e.g. following the mouse moves and clicks, eye-tracking, etc.). To ease our empirical research we created our own decision simulation software. It is designed to be flexible so that any data centric decision can be easily implemented. It shows to the decision maker the values of various data items related to the decision that needs to be made. It also logs the ones actually considered by the user. The decision maker is not provided with any guidance or help. There is only a 'calculator' tool that assists with the data derivations (such actions are also logged). The goal of the simulation is to pick one of the decision alternatives (displayed in a list within the software) based on the actual values shown in each scenario. An example of the interface is shown in Fig. 5. We call this kind of software 'decision-aware'. More details on the software and on how the logging is performed are available in [1].

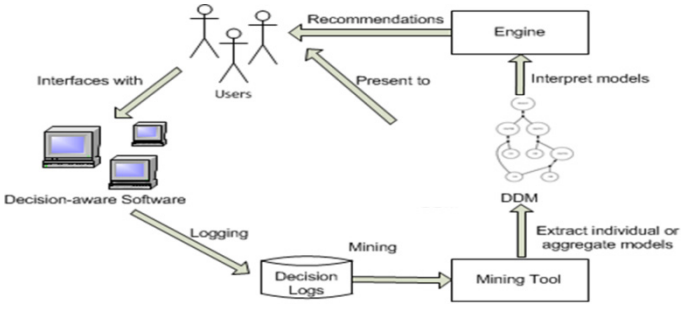

Fig. 2. The framework of decision making process mining

The logs are processed by a mining application that outputs individual or aggregated Decision Data Models. More details, on how the data is mined and the individual DDM is created, are available also in [1]. The aggregated model is the main focus of this paper. The aggregation can be done for all the traces in the log or just for a selection. Once an aggregated model is created, the process of one individual cannot be distinguished from the one of another individual. Also, in an

aggregated model, one cannot look at the most frequent path in the sense of a workflow model. But it is annotated with the frequency of each data item, so the most frequent behavior of the subjects is easily visible (see Fig. 7).

Clustering traces can be interesting because it provides useful insights into decision processes. Clustering can be done based on pre-determined particularities of decision makers or based on the similarity of the mined models. The similarity metrics and the clustering based on DDMs are not the focus of the current paper and will be available in a separate paper. But once a cluster is created, the aggregated DDM can be created to depict the behavior of the decision makers in it.

Once the DDM (individual or aggregated) is created it can be introduced, as it is, to the users. In the experiment introduced in section 4 we try to prove that a DDM is easier to read and understand than other workflow notations. It can be used to gain a better understanding of the actions of a particular decision maker or to look at the aggregated behavior of many individuals.

It is also possible to provide recommendations for actions to be performed during the decision process, based on a previously mined DDM. The goal is to guide a person through the steps that need to be performed in order to make an informed decision, up to the point where the choice of one alternative needs to be made. How the recommendation can be made, based on the DDM, is also not the focus of this paper and will be approached in a separate paper.

There are several limitations to our approach:

- our entire approach relies on the ability to log the activities (some of them mental) performed by the users of the software, during the decision process. The quality of the logs relies on the logging methods. We use direct and indirect methods. Direct methods rely on the direct observation of the decision maker (e.g. eye tracking). The indirect methods rely on the possibility to rebuild the mental activities of the decision maker from his actions. The key is to find and implement the tools that will force and enable the user to enact his every thought. The formal approach is created to support both types of logging. However, the case studies and experiments rely on decision-aware software that uses only indirect methods for logging;
- it is very important not to guide the decision maker in any way. This means, basically, that all the data should be provided in a single form and in a random manner. Of course, this is unfeasible. For example, in the implementation used for the case study, we show to the user two tabs, one with data related to buying a house and one related to renting a house, with some common data items available in both (e.g. the monthly income). Hopefully, it has a minimal influence on the decision process.

There are also several assumptions we are making:

- the decision scenario provided to the user contains all the data needed for the decision at hand. The user doesn't need any other essential information;
- since the user is not guided in any way, the actions performed, and their sequence, are a direct reflection of his mental process;
- the data provided in the scenario allows the user to explore and evaluate all the possible decision alternatives;
- A particular user might overlook certain aspects of a decision. But, all the aspects of a particular decision should be discoverable, if a large number of users are observed.

4 Case Study and Evaluation

This section aims to prove that our approach is feasible, and that new insight into the behavior of large numbers of decision makers is possible using a mined, aggregated DDM. The first two sub-sections also argue that the main process mining algorithms are not appropriate for extracting decision models based on logged user software interaction. The last sub-section introduces one of the experiments we performed in order to validate our claims.

4.1 Running Example and Comparison with Process Mining Algorithms

This subsection aims to provide a walk through our framework using a small running example. The user is presented with data regarding his financial position. He needs to decide if he can afford to buy a house. We assume a log of actions of three users, therefore obtaining three decision traces (DT1, DT2 and DT3 introduced in section 2).

The individual models for the three traces are shown in Fig. 3. We argue that the DDM clearly depicts how the data was used by the decision maker in order to reach a decision. For example, in DT1 the model shows that one action of the user was to aggregate the price of the house with the savings, producing the first derived data item (DD1). Then, given a possible loan period of 240 months, the minimum monthly installment is calculated (DD2) which is checked against the monthly income (DD3) to determine if it is affordable. Another check is performed to determine the minimum number of months needed to repay the debt (i.e. if all income was available).

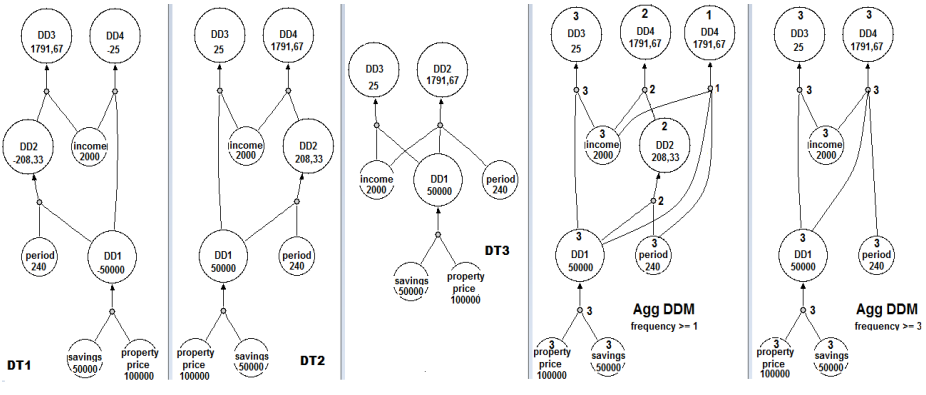

Fig. 3. The individual models of the three decisional traces introduced in section 2 (DT1, DT2 and DT3), the aggregated DDM showing all data elements and the abstracted aggregated DDM showing only elements with frequency of at least 3

The first aggregated DDM is created using the basic notion of derived element equivalence and Algorithm 1 (defined in section 2). For example, it is obvious that DT1.DD1 = DT2.DD1 = DT3.DD1 or that DT1.DD2 = DT2.DD2. But DT1.DD3 = DT2.DD4 ≠ DT3.DD2, therefore DD4 in the first aggregated DDM can be produced using an operation observed twice and DD5 one observed once. The aggregated DDM

showing data items with frequency greater than 3 uses extended element equivalence and Algorithm 2. As shown in Definition 8, if we use the extended equivalence notion, the input sets of DT1.DD3 = DT2.DD4 = DT3.DD2 = {DD1, income, period}. Therefore, the operation op3 = (DD4, 1761.67, [(+, income),(-, dd1), (/, period)], t3) is created and it is labeled as being observed 3 times.

The model in Fig. 4. is mined using Heuristics algorithm, one of the most popular process mining (PM) algorithms. There are various flaws in the logic of the process. For example, after the start activity both savings and property_price are enabled. A possible sequence is: savings, (property_price - savings), property_price. But this lacks logic because one cannot calculate if he can afford a house without knowing first the price of that house. This logic is used and captured in all the traces and is not shown by the model in Fig. 4. But it is enforced by the semantics of the DDMs in Fig. 3. And there are many such examples that prove, even on such small example traces, that the PM algorithms are not suitable for mining decision logs. This is because PM looks only at the names (labels) of the activities. In a mental decision process, the decision maker rarely assigns a name to a calculated value. It is close to impossible to automatically assign a name based only on logged data. Another cause is that, basically, PM looks at the sequence and the frequency of items in a log. The sequence is essential in a business process (e.g. one cannot 'process' a claim if it is not 'registered' first) while in decision making it is rather loose. For example, it is the same process the one in which a decision maker first finds out all the basic data items then performs calculations; with the one in which he finds some of the basic data, performs some calculations, realizes new basic data is needed and finds it out, then performs further calculations. The only condition required for similar decision processes is that the choice should be based on the same basic and derived data items.

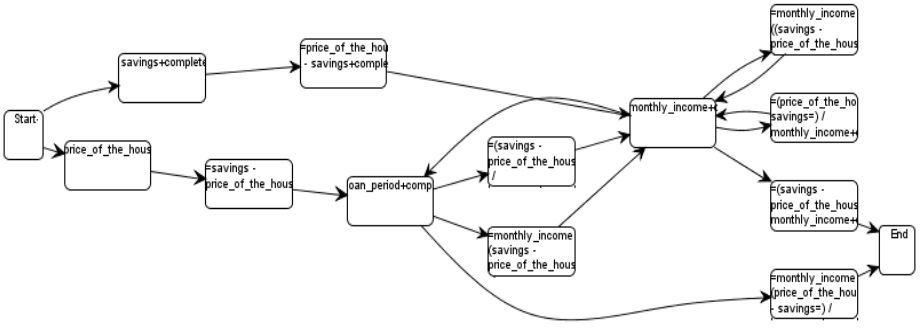

Fig. 4. The model mined with *Heuristics* Miner from DT1, DT2 and DT3

4.2 Case Study

In order to be able to experiment with our framework, we created a decision-aware system with several implementations. One implementation is simpler and introduces the user to a decision related to buying or renting a house. It only has 10 basic data items and two decision alternatives (to buy or rent the house). It is available at http://www.edirector.ro/v3_1/ (username "test" and password "test"). The second

implementation introduces the decision maker to the financial data of a company (balance sheet, cash flow, revenues and expenses, trial balance), the loan market (various data for six different loans) and one investment evaluation. The user needs to make decisions for the value of the loan to be contracted, the period, the type of the loan or the down payment to be made. This implementation is available at: http://edirector.ro/processmining_v2_en/ (also with username and password "test").

The interface of the implementation is shown in Fig. 5. The goal of Fig. 5 is to show how we mapped the formalism introduced in Section 2 to a software. It shows some basic data items related to the decision at hand (A), allows the user to input new data elements (B) and to derive data (C). The user is not guided in any way during the decision process. All the actions (clicks) performed by a user (which basic data items are used, how a derivation is performed, what new data is inputted, etc.) are logged.

There is one limitation of the current decision-aware software which is that all data items need to be numeric in order to be involved in data derivations. The software does not support data derivations based on qualitative data items.

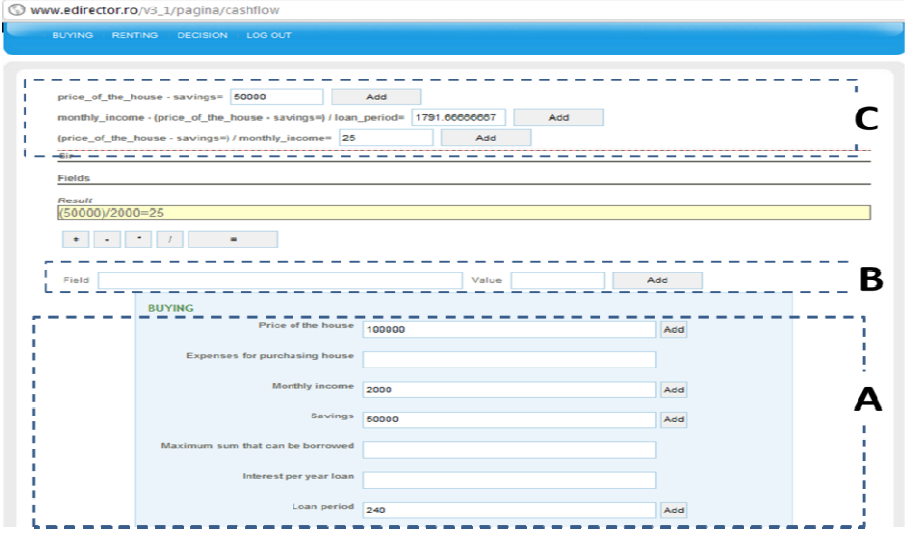

Fig. 5. Interface of the implemented decision-aware software, highlighting the *basic data (A)*, *inputted data (B)* and *derived data (C)* areas

In Fig. 6 we show a small portion of a log used in this case study. The entire log is available at http://edirector.ro/v3_1/export/pm.xml. The important records that are used by the mining algorithm are highlighted. The main purpose of this figure is to show that a derived data element (e.g. last highlighted record) is named in the log by the formula used to calculate it. Therefore if the same formula is used by two users, the labels of the two activities are similar in the two traces. But if the same derived data item is calculated differently (e.g. instead of A+B it is calculated as B+A) the labels will be different. More details on logging and logs are available in [1].

PI-ID	Value	Timestamp	ATE-ID	WFMElt	Name	Value
114	user10	11/20/2011 10:05:48:449	2574	click menu item	buying	buying
114	user10	11/20/2011 10:05:58:750	2575	click textbox	price_of_the_house	100000
114	user10	11/20/2011 10:05:59:924	2576	click buton	Add_price_of_the_house	price_of_the_house
114	user10	11/20/2011 10:06:01:667	2577	click buton	minus	-
114	user10	11/20/2011 10:06:03:318	2578	click textbox	savings	50000
114	user10	11/20/2011 10:06:04:205	2579	click buton	Add_savings	savings
114	user10	11/20/2011 10:06:05:554	2580	click buton	=price_of_the_house - savings	50000
114	user10	11/20/2011 10:06:31:296	2581	click textbox	monthly_income	2000
114	user10	11/20/2011 10:06:32:7	2582	click buton	Add_monthly_income	monthly_income
114	user10	11/20/2011 10:06:33:483	2583	click buton	minus	-
114	user10	11/20/2011 10:06:34:956	2584	click buton	Add_(price_of_the_house - savings=)	(price_of_the_house - savings=)
114	user10	11/20/2011 10:06:36:955	2585	click buton	impartire	/
114	user10	11/20/2011 10:06:39:88	2586	click textbox	loan_period	240
114	user10	11/20/2011 10:06:39:874	2587	click buton	Add_loan_period	loan_period
114	user10	11/20/2011 10:06:45:421	2588	click buton	=monthly_income - (price_of_the_house - savings=) / loan_period	1791.66666667
114	user10	11/20/2011 10:07:19:286	2589	click buton	Add_(price_of_the_house - savings=)	(price_of_the_house - savings=)
114	user10	11/20/2011 10:07:20:538	2590	click buton	impartire	/
114	user10	11/20/2011 10:07:22:121	2591	click textbox	monthly_income	2000
114	user10	11/20/2011 10:07:23:216	2592	click buton	Add_monthly_income	monthly_income
114	user10	11/20/2011 10:07:24:381	2593	click buton	=(price_of_the_house - savings=) / monthly_income	25

Fig. 6. Partial trace from the logged user actions

The DDM created using both Algorithm 1 and Algorithm 2 is shown in Fig. 7. It is an abstracted model that shows only data elements with frequency greater than 2. This greatly reduces the size of the model because there are many items that have a frequency of 1. We believe that if a data element shows up once, it is exceptional behavior which can be safely abstracted from. Items with frequency of 2 are mostly outliers that show up in the same trace. The frequency of a data element is the number shown above its name. Because there are no derived data elements produced by two different operations, the frequency of an operation is the same as the frequency of the produced derived data element (therefore it is hidden in this particular model). To encourage the reader to follow the main behavior, the most frequent data elements are highlighted.

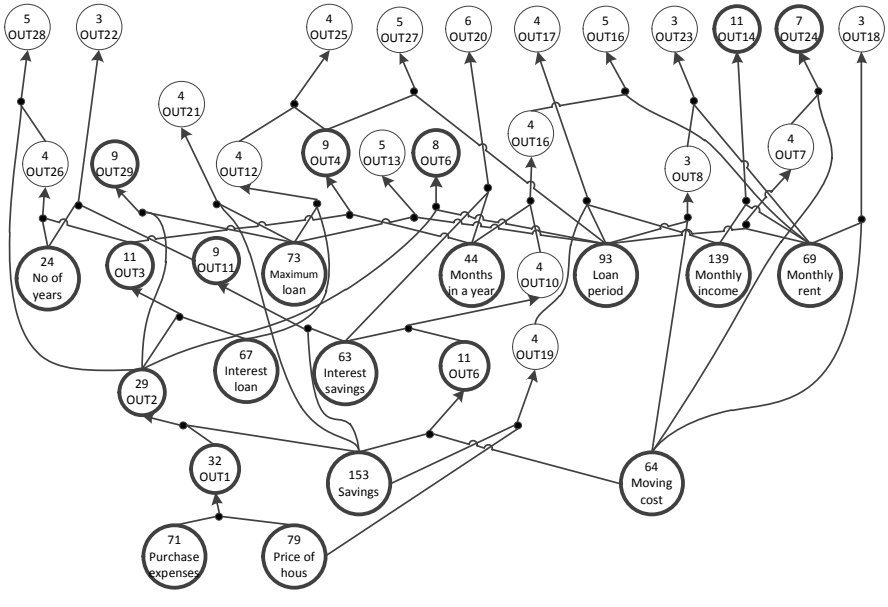

Fig. 7. Aggregated DDM (frequency of data elements greater than 2)

As one can notice, the size of the model is quite small at 40 elements (we use as a reference the conclusions presented in [2]). It is also interesting to mention that the size of the model remained relatively stable after 25 of the traces were aggregated (it already had 35 elements). This allows us to argue that, even if another 50 traces were aggregated, there is a below linear increase in the size of the model if a frequency threshold of at least 2 is used. This cannot be said about the process mining algorithms we used as a comparison. The Alpha, Heuristics and Genetic Algorithm models are spaghetti like, therefore unusable (due to the lack of space we cannot show them in this paper). Also, the size increase from 25 to 50 traces is noticeable. This is mainly due to the new behavior that is added and because of the naming of the derived data elements. The Fuzzy model is readable due to the zoom in/out feature. However, the model (for this example) does not show any new insights mostly because activities in the clusters are not logically connected.

Based on this aggregated model, is possible to extrapolate the main behavior of the decision makers. They first find out if the house can be purchased from savings (element OUT2 with frequency 29). Since this is not possible, they find out if the purchase can be afforded by using a loan (OUT29). Since this is possible they calculate the monthly installment of capital (OUT4) and interest (OUT5). Some of the users compare the monthly rent with the monthly income (OUT14) while others also take into account that the savings (in case the money is not used for the purchase) will fetch a monthly interest (OUT11).

If we take a closer look at some of the infrequent elements we can see that about 10% of the users were also interested in how much rent they would pay for the entire period (OUT7). This is a part of an alternative path to reaching the final decision.

Also, it is noticeable that some of the basic data elements are more important (more frequently used) than others. It is possible to say, for example, that the most important piece of data, for this decision, is the monthly income.

Because of the abstraction one can notice that some of the input derived elements are less frequent than the derived elements (e.g. look at OUT7 and OUT24). This is due to the fact that the alternative operations were less frequent than the threshold).

There is also no root (final decision). If we place a root in the model and want to keep the semantics of DDM, there should be alternate operations producing the root that show which are the data items used to make the actual decision (and use the frequency to show if the same operation is used by multiple decision makers). But it is a difficult task to log the actual criterions that are used to make the final decision, without asking the users to explicitly point them out.

4.3 Experiment

To evaluate if a DDM is understandable, we designed an experiment that asks the subjects to follow the steps prescribed by the model. The point we try to prove is that: "A DDM is easy to read and understand and provides better insights into the decision process than other knowledge representations (YAWL or C-nets)".

To provide the answer to the statement, we created a balanced single factor experiment with repeated measurements. We investigated the effects of a single factor (a particular model type) on a common response variable to prove the degree to which the subjects understand the model. The measured response variable is two output

values that should be calculated by strictly following the model. The correctness of the output values prove if the model was correctly interpreted by the subjects.

During the experiment, the members of the focus group had to calculate a value by following a printed process model. There were three different types of models, equivalent to each other. The experiment had 3 rounds so that each subject had to use each of the three models.

Some of the concerns that limit the validity of the results are:

- the subjects performed the same process three times, using different depictions of it. Of course that, by the third use, the subjects got to know the process. Therefore, how much of the model was used in the last round is questionable.
- subject's knowledge of each model may be different. The subjects must know and have similar experience with each model. To mitigate this risk we used as subjects master students that were given a-priori lectures on workflow models. Even more, the focus was on the YAWL and C-net models and less on the DDMs.
- the domain knowledge may influence the degree to which the subjects rely on the process models. To mitigate this risk, the scenario we used requires above average knowledge of accounting and the subjects had only basic knowledge on this subject.

We are aware that the results are not statistically valid because of the limited number of subjects involved in this first experiment. We used 12 master students at our Faculty, so that in each round there were 3 groups of four students. We believe that, even so, the findings are interesting and worth mentioning. The main result is that none of the students were able to derive the expected output values. The main conclusion that arises from this experiment is that, in order to make the DDM available for non-experts in a field, the exact operations need to be made available, somehow, to the decision makers. There was also a questionnaire that, among other questions, required the subjects to indicate their favorite model. Interestingly, 9 out of 12 indicated they prefer the DDM.

5 Related Work

As stated in the introduction, this paper starts with the assumption that the logs are available. Therefore, we focus on the models that can be created by mining those logs. We draw inspiration from the process mining field, which is concerned with extracting a model from logged activities [3]. The most important process mining algorithms are: Alpha [2], Heuristics [4], Genetic [5] and Fuzzy [6]. Some of those algorithms apply various solutions for dealing with less structured or noisy logs. But, as we showed in section 4, the specifics of human decision making activities don't fit the basic assumptions used by process mining algorithms (e.g. that there is a 'mainstream' process in the logs). There are also limitations given by the underlying semantics of the extracted models. This is why a new model, the C-net is emerging [3]. The declarative approach to process mining [7] also didn't work on decision logs.

The model we create uses the approach in Product-based workflow support, especially the semantics of Product Data Model (PDM) [2]. There are differences due to the main purpose of the model (a PDM is a general, design model while a DDM is

focused on showing the decision process as it is captured in the logs). There are also differences in the definition of a DDM (e.g. there is no root; each data element has a value, etc.).

The classical data mining approach that looks at extracting association rules (as first introduced in [8] and then improved by many papers) or the more actual web mining [9], don't fit this particular research because our goal is to extract a workflow perspective of the decision process.

The DDM is often regarded as a decision tree. However, the semantics of the two models are quite different (e.g. a DDM may have the same leaf node connected to many other and a derived data item can be produced by several alternative operations). Before arriving to the DDM, we explored the knowledge representation area (more specifically the ontologies). Even if we were happy with the expressiveness and the tools available, we identified two major problems. It is difficult to extract, build and maintain an ontology for each decision problem and it is a challenge to adapt it to the fuzzy processes performed by many individuals [10].

In decision making theory field could not find any approach over decision making as a workflow or some bottom-up approach that starts from the instance decision processes and seeks to aggregate them. From this area, one of the approaches that influenced us is [11] because it places human decision making at the intersection of decision theory, artificial intelligence and information systems and behavioral sciences.

6 Conclusions

The user interaction logs with software or web pages are a great source of knowledge. In this paper we introduced our approach aimed at creating an aggregated model of the data perspective of business decision processes. The framework requires a software to log the actions performed by users in the decision making process. Then, the logs are mined for the data view of individual decision making process (producing a Decision Data Model). This paper shows how individual models can be aggregated into a single model. The approach can be exploited in connection to business decisions since most of them are data-centric.

We formalized the essential notions that are used. A decision log needs to explicitly show a causal relation (dependency, connection) between some inputs and some outputs. The DDM graphically depicts those relationships using notions as basic/derived data elements and operations. An aggregated model merges instance DDMs using the notions of basic or extended data element equivalence for mapping identical nodes to each other. It is also possible to create an aggregated model where the frequency of elements (nodes and operations) is above a certain threshold.

We showed how our approach works on a running example. We also demonstrated that there are particular features of decision processes that render existing process mining algorithms useless. For experimenting and validating our approach we used simulation software (i.e. we created an artificial decision problem and we supplied data, ranging from critical to trivial, for solving it). We randomly picked 50 traces from the log to show that our approach produces a readable and usable model that provides insights into how the users made the decision of buying or renting a house.

Logging user actions can easily be replicated to any system that provides data for decision making (e.g. ERP systems, on-line stock trading, etc.) either by following mouse movements or by employing eye-tracking. The logging issue limits our output only when new information is provided to the decision maker outside the boundaries of the software. This limitation can be trivial for real life situations that are computer-centric (e.g. stock trading) while it can render it useless for others (e.g. negotiations).

Applying our approach to real life situations is straight-forward. The only check that needs to be done before using it right away is whether the available real-life basic data is also included in the DDM. If there are more data items in real life, the user needs to evaluate the relevance of the extra data.

We do not claim to have found a solution for any kind of decision processes. It is our goal to create a domain-independent approach that can be applied with minor adjustments for various data-centric decisions. Still, every human being is unique and our method also tries to abstract from the personal features (e.g. patience in solving problems, IQ, etc.). We feel that there is a strong connection between the decision maker's human model and the individual DDM but it is all lost when aggregating tens or hundreds of individual models.

The paper is focused on the aggregation of models. Our decision-process mining framework also allows other kind of analysis such as clustering decision makers; comparisons and emphasis of the differences between two individual models, or comparisons between an individual model and an aggregated one.

We showed one of the validation experiments but there is more work to be done on this issue. We need to extend this particular experiment by involving larger numbers of subjects, with various backgrounds, so we can achieve statistically relevant results. We will also use more process notations and knowledge representations (e.g. BPMN, rule-based descriptions).

Acknowledgments. This work was supported by CNCS-UEFISCSU, project number PN II-RU 292/2010; contract no. 52/2010.

The author would like to thank Wil van der Aalst and Irene Vanderfeesten for sharing their ideas and providing valuable feedback which steered our entire research, not just the part presented in this paper.

References

1. Petrusel, R., Vanderfeesten, I., Dolean, C.C., Mican, D.: Making Decision Process Knowledge Explicit Using the Decision Data Model. In: Abramowicz, W. (ed.) BIS 2011. LNBIP, vol. 87, pp. 172–184. Springer, Heidelberg (2011)
2. Vanderfeesten, I., Reijers, H.A., van der Aalst, W.M.P.: Product-based workflow support. J. Information Systems 36, 517–535 (2011)
3. van der Aalst, W.M.P.: Discovery, Conformance and Enhancement of Business Processes. Springer, Heidelberg (2011)
4. Weijters, A.J.M.M., Ribeiro, J.T.S.: Flexible Heuristics Miner (FHM). In: IEEE Symposium on Computational Intelligence and Data Mining (CIDM), pp. 310–317. IEEE Press, New York (2011)

5. de Medeiros, A.K.A., Weijters, A.J.M.M., van der Aalst, W.M.P.: Genetic Process Mining: An Experimental Evaluation. Data Mining and Knowledge Discovery 14(2), 245–304 (2007)
6. Günther, C.W., van der Aalst, W.M.P.: Fuzzy Mining – Adaptive Process Simplification Based on Multi-perspective Metrics. In: Alonso, G., Dadam, P., Rosemann, M. (eds.) BPM 2007. LNCS, vol. 4714, pp. 328–343. Springer, Heidelberg (2007)
7. Pesic, M., Schonenberg, H., van der Aalst, W.M.P.: DECLARE: Full Support for Loosely-Structured Processes. In: Eleventh IEEE International Enterprise Distributed Object Computing Conference (EDOC 2007), pp. 287–298. IEEE Computer Society (2007)
8. Agrawal, R., Imielinski, T., Swami, A.: Mining Association Rules Between Sets of Items in Large Databases. In: ACM International Conference on Management of Data (SIGMOD 1993), pp. 207–216. ACM Press, New York (1993)
9. Liu, B.: Web Data Mining Exploring Hyperlinks, Contents, and Usage Data. Springer, Heidelberg (2011)
10. Fensel, D.: Ontologies: a Silver Bullet for Knowledge Management and Electronic Commerce. Springer, Heidelberg (2001)
11. French, S., Maule, J., Papamichail, N.: Decision behavior, analysis and support. Cambridge University Press, Cambridge (2009)