



Discovering Causal Relations in Semantically-Annotated Probabilistic Business Process Diagrams

Hector G. Ceballos and Francisco J. Cantu

Tecnologico de Monterrey, Monterrey, 64849, Mexico,
ceballos@itesm.mx, fcantu@itesm.mx

Abstract

Business Process Diagrams (BPDs) have been used for documenting, analyzing and optimizing business processes. Business Process Modeling and Notation (BPMN) provides a rich graphical notation which is supported by a formalization that permits users automating such tasks. Stochastic versions of BPMN allows designers to represent the probability every possible way a process can develop. Nevertheless, this support is not enough for representing conditional dependencies between events occurring during process development. We show how structural learning on a Bayesian Network obtained from a BPD is used for discovering causal relations between process events. Temporal precedence between events, captured in the BPD, is used for pruning and correcting the model discovered by an Inferred Causation (IC) algorithm. We illustrate our approach by detecting dishonest bidders in an on-line auction scenario.

1 Introduction

Organizational processes have multiple representations, being flowcharts, workflows and causal diagrams the most common and intuitive. Among them, Business Process Modeling and Notation (BPMN) has become the most accepted representation among industry practitioners due to its rich visual expressiveness for representing complex workflows where multiple actors decide which alternative to follow in order to reach individual or global goals [1].

BPMN Business Process Diagrams (BPDs) basically describe a process in terms of events and actions (nodes) connected through control flows (arcs) that indicate valid sequences in the process development. Gateways are special nodes connected through control flows that indicate whether the process develops through two or more sequences of task and events in parallel (AND), alternatively (XOR) or optionally (OR). The beginning of the process is denoted by an initial event node and its conclusion by a set of end event nodes.

Gateways allow to express all the possible ways a process can develop but the probability of observing one or another is not represented in the original version of BPMN. Stochastic versions of BPDs have been proposed for estimating the probability of observing a given event or task at time t . For this purpose tasks are annotated with a duration interval [2], whereas alternative sequence flows are annotated with a probability [3]. Both approaches use Continuous Time Markov Chains for representing BPDs.

Another approach is to generate a Bayesian Network (BN) from events occurring in a BPD [4]. Probabilities of the resulting network are learned from past process executions and can be used for making Bayesian inference, i.e. estimating how likely it is to observe an event or a task B given that another one A occurs, even if B does not follow immediately to A .

In this paper we evaluate a structural learning algorithm for discovering causal relationships between events occurring in past process developments. Besides, we use semantic annotation of BPD nodes for extracting a probabilistic distribution of a subset of process instances. This is illustrated by estimating if a bidder will pay an item awarded in an on-line auction, given the reviews of previous auctions.

In the next section we present the probabilistic BPMN normal form and how to annotate BPD nodes semantically. In section 3 we introduce a new semantic descriptor and present how Bayesian structural learning is used for discovering casual relationships and use them for detecting fraudulent bidders. We conclude by discussing the relevance of this work and providing closing remarks.

2 Background

The proposed algorithm is based in a BPMN probabilistic normal form that can generate a Bayesian Network that transforms precedence relations into conditional dependencies. For this purpose, we now provide the required background.

2.1 Probabilistic BPMN normal form

In the probabilistic BPMN normal form introduced by Ceballos et al [4], a Business Process Diagram \mathbf{W} is formally represented by a set of pools (\mathbf{P}), lanes (\mathbf{L}), nodes (\mathbf{N}) and control flows (\mathbf{F}). Nodes (\mathbf{N}) allowed in this probabilistic normal form are: start events (N^S), intermediate events (N^I), end events (N^E), atomic actions or tasks (N^A) and gateways (N^G). All sequence flows are unconditional, denoted as $F(n_i, n_j) \in \mathbf{F}$ where $n_i, n_j \in \mathbf{N}$. A single start event $s \in N^S$ is defined ($|N^S| = 1$).

Additionally, two consecutive action nodes must be mediated by at least one intermediate event node and as many gateways as needed, i.e. two action nodes are not connected directly through sequence flows. Each split or merge of control flows must be mediated by a splitting gateway ($N_S^G \subseteq N^G$) or a merging gateway ($N_M^G \subset N^G$), respectively. Gateways can be of type Parallel-AND (A), Optional-OR (O), or Exclusive-XOR (X). The graph G_N constituted by all $F(n_i, n_j) \in \mathbf{F}$ must not have any directed cycle or loop.

For obtaining a Bayesian Network from a normalized BPD, events and actions are mapped to realizations of random variables, whereas control flows and gateways are used for building the conditional dependency graph and the probabilistic distribution of the model.

A Bayesian Network (BN) is represented by $M = \langle V, G_V, P(v_i|pa_i) \rangle$, where V is the set of random variables, G_V is the Directed Acyclic Graph (DAG) consisting of variables in V and directed arcs between them, and $P(v_i|pa_i)$ is a conditional probabilistic distribution (CPD) where the probability of v_i depends on the value of its parents (pa_i) in G_V . The set of possible values a discrete random variable may hold is denoted by its *domain* $dom(V_i) = \{v_{i1}, \dots, v_{in}\}$, whereas the *realization of a random variable* V_i to the value $v_{ij} \in Dom(V_i)$ is represented as $V_i = v_{ij}$.

Task nodes as well as start, end and intermediate events are mapped to realizations of random variables. This mapping, denoted by $map(n, V_i = v_i)$, $n \in \mathbf{N}$ where $n \in (\mathbf{N} \setminus N^G)$, is produced during the BN generation.

Given that control flows in the BPD encode necessary conditions for the development of the process, they are interpreted as temporal precedence and conditional dependence in the resulting BN. A graph between BPD mapped nodes ($G'_N : N \times N$) is modified according to a set of rules that remove unnecessary gateways and unify end nodes in a single one. The resulting graph contains the conditional dependencies between random variables.

Figure 1 shows the structures supported by this probabilistic normal form. The column Mappings shows the correspondence between BPDs and random variables whereas the last column indicates which mapped nodes prevail in G'_N .

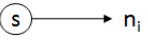


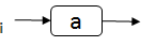
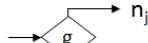


Structure		Constraint	Mappings	In G'_N
Trigger		$n_i \in N \setminus N^S$	$\text{map}(s, Z_s=\text{True})$	s
Outcome		$n_i \in N \setminus N^E$	$\text{map}(e, Z_e=e_i)$	e_1
Event		<i>i.</i> $n_i \notin N^G$ <i>ii.</i> $n_i \in N^G \wedge \neg \text{type}(n_i, X)$	$\text{map}(i, Z_i=\text{True})$	i
Actions		$n_i, n_j \in N \setminus N^A$	$\text{map}(a, X_a=\text{True})$	a
Split gateways	Decision 	$\text{type}(g, X), n_i \in N,$ <i>i.</i> $n_j \in N^I$ <i>ii.</i> $n_j \in N^G \wedge \text{type}(n_j, X)$	$\text{map}(n_j, Z_g=n_j)$	g
	Alternative 	$\text{type}(g, A) \vee \text{type}(g, O)$ $n_i, n_j \in N$	n_i, n_j	n_i, n_j
Merge gateways		$n_i, n_j \in N$	n_i, n_j	n_i, n_j

Figure 1: Valid structures in the BPD normal form.

2.2 Semantic Descriptors

Semantic descriptors of BPDs introduced in [5] use the notation and properties of conjunctive queries given by Description Logics (DL) [6], which states that the interpretation of a query is not only given by statements specified in the query, but by constraints and definitions specified in the domain model (\mathcal{T}) as well. This model, known as *ontology*, is basically constituted by a set of *concepts* (used to group/classify objects) and *relations* (used to specify entity's attributes or used to relate entities among them). Ontology languages such as OWL¹ allows expressing concepts in terms of other concepts (definitions), and specifying constraints between concepts (e.g. disjointness or subsumption) and properties (e.g. reflexivity or transitivity).

A DL *conjunctive query* (CQ) has the form:

$$Q = (s_1, \dots, s_n). \{T_1, \dots, T_m\}$$

¹OWL Web Ontology Language. <https://www.w3.org/TR/owl-features/>

where \mathbf{s}_i is the set of distinguished variables, denoted $Dis(Q)$, that define the resulting binding sets (the information retrieved), and T_i is a finite set of either *concept clauses* ($\mathbf{s} \text{ rdf:type } \mathbf{C}$) or *relation clauses* ($\mathbf{s} \text{ r } \mathbf{s}'$), where $\mathbf{s}, \mathbf{s}' \in (N_V \cup N_C)$, \mathbf{C} is a concept/class and \mathbf{r} is a relation/property (both defined in \mathcal{T}), N_V is a finite set of variables denoted $Var(Q)$, and N_C is a finite set of constants.

In SPARQL, the query language for RDF [7], a conjunctive query is coded as a SELECT query without filters. Distinguished variables ($Dis(Q)$) of a conjunctive query Q are listed in the section SELECT, whereas atoms T_i are specified in the section WHERE as statements or subject-predicate-object triplets.

The semantic descriptor of BPD nodes is defined as follows.

Definition 1 (Semantic Descriptor). *A semantic descriptor $Ann(n, Q)$ uses the conjunctive query Q for representing the meaning of a lane, an observable event, or human task $n \in (\mathbf{L} \cup N^S \cup N^I \cup N^E \cup N^A)$ in a BPD \mathbf{W} .*

A *lane descriptor* describes the kind of individual that play a role in the activity. It has the form $Ann(l, Q_l)$, where $l \in \mathbf{L}$ and Q_l might represent an absolute or relative role. An *absolute role* annotation is given by a $Q_l = (?role).\{?role \text{ a } RoleClass\}$ where **RoleClass** indicates the type of individual, denoted by $?role$, associated to l . A *relative role* annotation is given by a $Q_l = (?role).\{?role \text{ rel } ?role2\}$ where the role associated to the lane ($?role$) is defined in terms of its relationship (**rel**) with a participant represented by another lane ($?role2$).

In this formalization, an *event descriptor* has the form $Ann(z, Q_z)$, where $z \in (N^S \cup N^I \cup N^E)$ and Q_z is a conjunctive query describing a condition (constraints between individuals) that denotes the occurrence of the event. Formally, the occurrence of the event z is observed in the Linked Data \mathcal{K} by evaluating Q_z , i.e. $\mathcal{K} \models_{\pi} Q_z$, where \mathcal{K} contains information of past process developments and π contains references to entities participating in event occurrence(s). Similarly, a *task descriptor* describing the task execution has the form $Ann(x, Q_x)$, where $x \in N^A$. In consequence, a BPD annotated semantically is defined as follows.

Definition 2 (Annotated BPD). *An annotated BPD $\mathcal{W}_{\mathcal{D}}$ is a BPD \mathcal{W} where each node $n \in (\mathbf{L} \cup N^S \cup N^I \cup N^E \cup N^A)$ is annotated with exactly one semantic descriptor $Ann(n, Q) \in \mathcal{D}$.*

3 Discovering causal relations in Business Process Diagrams

In this work we extend the semantic annotation proposed in [5] with a new semantic descriptor for making inference on subsets of process instances. We also introduce a hybrid probabilistic-semantic representation of past process developments that will be further used for learning and decision making.

We motivate our approach with a case study in an on-line auction scenario. The BPD shown in Figure 2 is an adaptation of an English on-line auction processing modeled by Kuster et al in [8], compliant with the probabilistic BPD normal form proposed by Ceballos et al in [4]. The process starts when the auction is over, then the item is awarded to the highest bid or is declared void on the absence of bids. The awarded bidder is notified by the auctioneer and he makes the corresponding money transfer. The seller receives the payment notification and sends the goods to the bidder. The auctioneer transfers the money to the seller if the item is sent or returns it back to the bidder otherwise. Simultaneously, both bidder and seller rate the auction. Finally, the process ends when the auction is removed by the auctioneer from the system, indicating whether the item was sold or not.

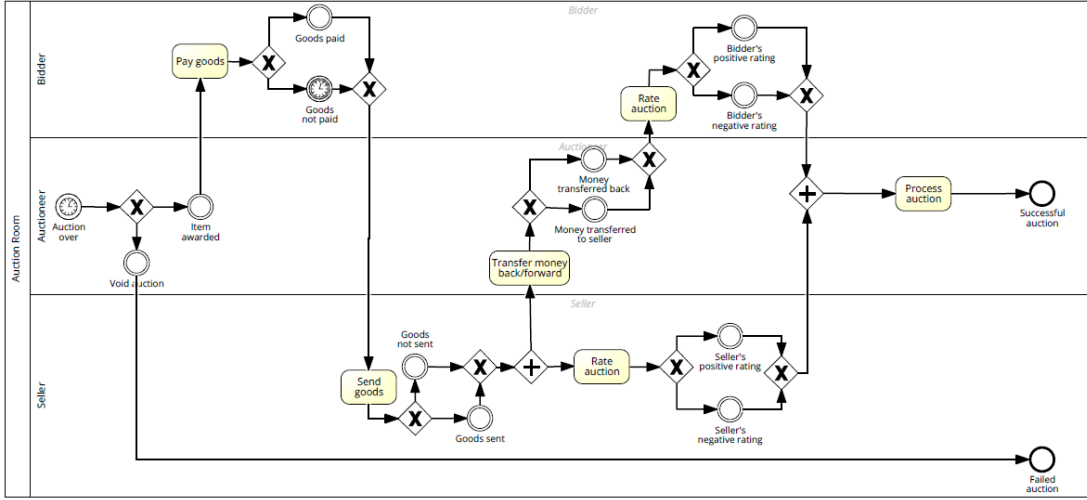


Figure 2: BPMN diagram of the processing of an English electronic auction.

Figure 3 shows the conditional dependence graph of the Bayesian Network resulting of removing tasks and applying the transformation described in [4] to the BPD in Figure 2. The link $V_{BID} \rightarrow V_{AP}$ is used for representing that the process ends immediately in void auctions, denoted as $V_{AP} = FAILURE$. Items sold in auctions are captured by $Ann(V_{AP} = SUCCESS)$. Descriptors of V_{REVB} and V_{REVS} indicate how both participants are evaluated, denoted by the semantic variables $?seller$ and $?bidder$; these semantic variables refer to the same individuals ($?a, ?seller, ?bidder$) used in the descriptor of the start event (V_{AO}).

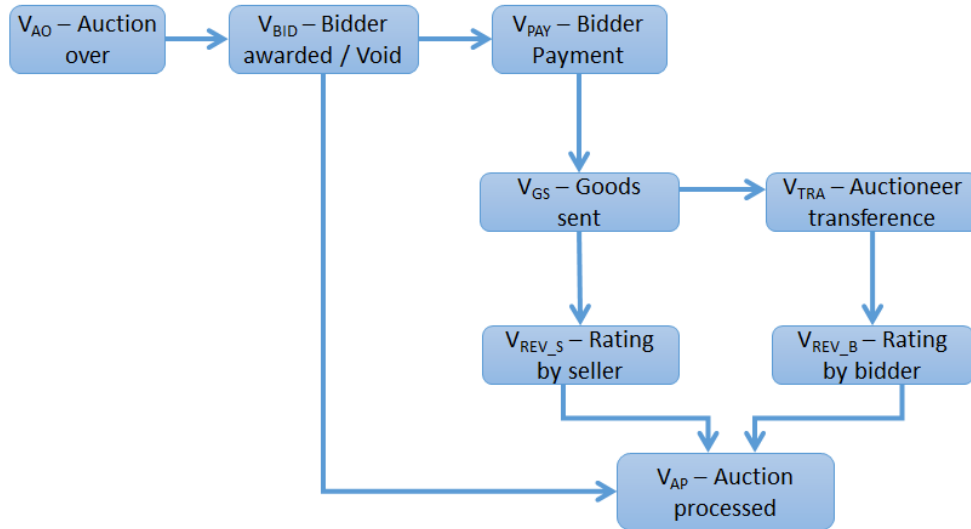


Figure 3: Bayesian Network for the on-line auction processing.

Table 1: Semantic description for identifying the awarded bidder (V_{BID}).

No.	V_i	v_{ij}	Q_{ijl}
1	V_{BID}	?bidder	$Q_{BID}() = (?bidder).\{?a \text{ rdf:type : Auction .}$ $?a : \text{awardedBidder ?bidder } \}$

3.1 Extending semantic annotation of BPDs

In [5], semantic descriptors are used for realizing random variables to a finite set of values given in their domain. Adjusting the probabilistic model to values found in the knowledge base \mathcal{K} from which process instances are recovered requires making a bridge between both semantic and probabilistic representations, as it is done in other probabilistic-logic approaches [9, 10, 11]. For this reason we introduce the following descriptor type.

Definition 3 (Bridge Descriptor). *A bridge descriptor is a semantic descriptor $Ann(n, Q_B)$ such that its conjunctive query Q_B has a single distinguished variable s_B , i.e. $Dis(Q_B) = \{s_B\}$.*

In Figure 1 it can be observed that task nodes, as well as start and intermediate events are mapped to a realization $V_i = True$. The purpose of the bridge descriptor is to introduce a value taken from the process instance (stored in \mathcal{K}), in the conditional probabilistic distribution of the BN.

For instance, if we are interested in knowing the behavior of a bidder in previous auctions we need a random variable which domain is linked to every single bidder, in order to condition the probability of other events on it. Let us define the random variable V_{BID} with the semantic description given in Table 1.

In this example, the domain of V_{BID} must contain the symbol *?bidder*, representing that V_{BID} can be realized to any symbol matching Q_{BID} in \mathcal{K} . New symbols pulled out from \mathcal{K} can be added to $Dom(V_{BID})$ during further Bayesian parametric learning phases.

3.2 A probabilistic and semantic representation of past process developments

For capturing the conditional probabilistic distribution of the process we need a formalism that represents instances of the underlying Bayesian Network obtained from a DL knowledge base \mathcal{K} , i.e. a Linked Data containing information of past process developments.

Definition 4 (Process instance). *An instance of a process modeled through an annotated BPD $\mathcal{W}_{\mathcal{D}}$ and compliant with a probabilistic normal form [4] is represented by a tuple $I_{\mathcal{W}} = \langle \pi_i, \bar{v}_i \rangle$ where π_i is a binding set identifying the symbols $c_i \in \mathcal{K}$ associated to semantic variables s_i , and \bar{v}_i stores a set of realizations of random variables $V_i \in V$ defined in the BN generated from $\mathcal{W}_{\mathcal{D}}$.*

A process instance $I_{\mathcal{W}}$ is well-formed if for any two bindings of semantic variables $(s_1 : c_1) \in \pi_i$ and $(s_2 : c_2) \in \pi_i$, $s_1 \neq s_2$, i.e. for each semantic variable s_i exists a single realization in π_i . Likewise for any two realizations of random variables $(V_1 = v_1)$ and $(V_2 = v_2)$ in \bar{v}_i . In summary, \bar{v}_i represents the occurrence of events represented by random variables V_i , whereas π_i keeps track of individuals or objects participating in these events.

Table 2: Joint probability distribution of $\mathcal{M}_{Auction}$.

Cases	AO	BID	PAY	GS	TRA	REV_B	REV_S	AP	Comment
5%	TRUE	FALSE	FALSE	FALSE				FAILURE	Void auction
4%	TRUE	?bidder	FALSE	FALSE				FAILURE	Bidder doesn't pay item
4%	TRUE	?bidder	FALSE	FALSE			NEG	FAILURE	Bidder doesn't pay item. Seller rates negatively.
5%	TRUE	?bidder	TRUE	FALSE	BIDDER	NEG		FAILURE	Bidder pays, Seller doesn't send the item. Bidder rates negatively.
65%	TRUE	?bidder	TRUE	TRUE	SELLER	POS	POS	SUCCESS	Item sold. Rating POS-POS.
8%	TRUE	?bidder	TRUE	TRUE	SELLER	NEG	POS	SUCCESS	Item sold. Rating NEG-POS.
6%	TRUE	?bidder	TRUE	TRUE	SELLER	POS	NEG	SUCCESS	Item sold. Rating POS-NEG.
3%	TRUE	?bidder	TRUE	TRUE	SELLER	NEG	NEG	SUCCESS	Item sold. Rating NEG-NEG.

An example of a void auction would be represented by the process instance $I_{Auction_void} = \langle \pi_{void}, \bar{v}_{void} \rangle$, where

$$\pi_{void} = \{?a : \langle things/auction_003 \rangle, ?seller : \langle people/john \rangle\},$$

$$\bar{v}_{void} = \{V_{AO} = TRUE, V_{BID} = FALSE, V_{AP} = TRUE, V_{PAY} = FALSE, V_{GS} = FALSE\}.$$

The value of the rest of the random variables ($V_{TRA}, V_{REV_S}, V_{REV_B}$) would be unknown, i.e. unobserved.

The procedure for the generation of process instances from a knowledge base \mathcal{K} is out of the scope of this paper.

3.3 Structural Learning

A conditional dependence graph does not constitute an independence map (I-Map) as long as the absence of arcs between variables does not necessarily represent conditional independence. In order to have a proper Bayesian Network that enables probabilistic inference we need to observe past process developments, complement the conditional dependence graph with missing conditional dependencies derived from observed patterns, and learn the parameters (CPD) of the resulting network.

Past process instances can be used for learning the probabilistic distribution of the process which in turn can be analyzed with a Bayesian classifier implementing an Inferred Causation (IC) algorithm for discovering additional conditional dependencies. This algorithm detects causal dependencies between variables and represents confounders (a third unobserved cause) through undirected arcs between variables. In this sense, the original conditional dependence graph and the partial order of random variables in the BN are used for redirecting and solving arcs. We illustrate this process next.

Table 2 shows an example of the joint probability distribution of the Bayesian Network obtained from the auction process. The probabilistic distribution is constituted by a set of valid process instances where the first column indicates the probability of finding such instance. Columns are labeled with the random variable sub-index and the last column provides a brief description of every case. Empty cells represent unobserved values for each case.

We generated a sample of 1,000 instances according to the distribution given in the Table 2. Using parametric learning on the original DAG (see Figure 3) only 87% of the instances were classified correctly.

Applying the IC search algorithm [12] implemented in Weka [13] with a BayesNet classifier we obtained 9 arcs (see Figure 4.a), 7 directed and 2 undirected (shown as bi-directed arcs). 5 of them (56%) were new viable edges, including the two undirected arcs. Two arcs were proposed in the opposite direction by ICS and the other two were discarded given that they would introduce directed cycles in the original DAG. The first undirected edge was solved by precedence constraints ($V_{PAY} \leftrightarrow V_{REV_S}$), and the other was solved using the partial order chosen for random variables ($V_{REV_S} \leftrightarrow V_{TRA}$).

To obtain an independence map (I-Map) we enriched the original DAG with five new valid causal dependencies discovered by ICS (see Figure 4.b): $V_{BID} \rightarrow V_{REV_S}$, $V_{PAY} \rightarrow V_{REV_S}$, $V_{PAY} \rightarrow V_{AP}$, $V_{TRA} \rightarrow V_{REV_S}$ and $V_{TRA} \rightarrow V_{AP}$. These five arcs can be added without introducing directed cycles in the graph.

We also built a third compact model using only arcs proposed by ICS (see Figure 4.c): 1) the five arcs identified as new and valid, and 2) the two arcs identified by ICS in opposite direction. The last two arcs were reversed for satisfying precedence constraints.

The classification result was tested with 10 folds cross-validation where 80% of the sample was used for training and 20% was used for testing, selected randomly. 100% of the cases were classified correctly in both enriched and compact models. For this example, both models constitute I-Maps that can be used for predicting the process outcome in future auctions. The probabilistic distribution (CPDs) of both models is obtained applying parametric learning with the given sample.

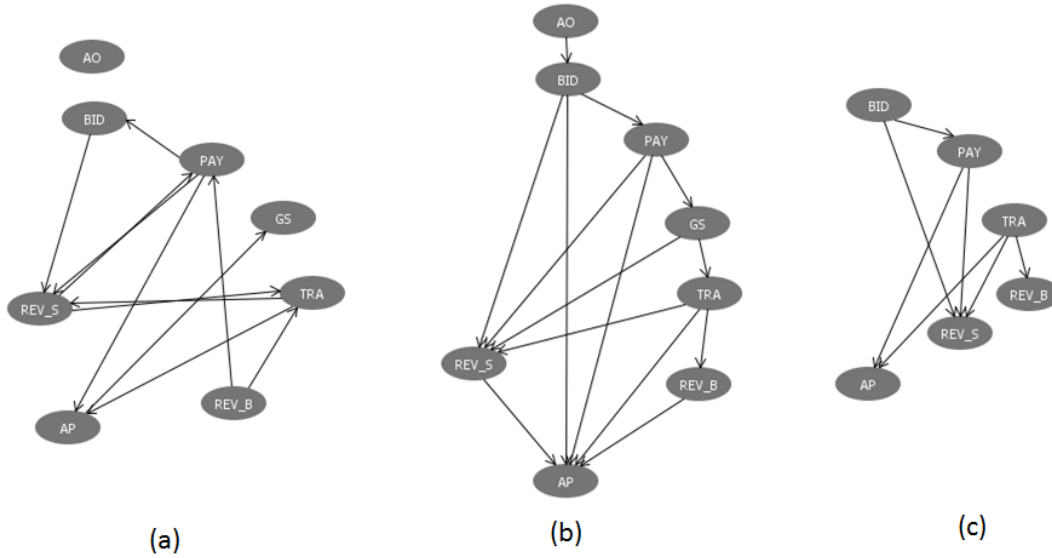


Figure 4: Networks for $\mathcal{M}_{Auction}$: a) the bi-directed graph obtained by ICS, b) the original DAG enriched with ICS valid arcs, and c) the ICS subgraph consistent with the original DAG.

3.4 Decision Making

Now we use semantic descriptors for determining probabilistically whether to award or not a given bidder based on his participation in previous auctions. In order to construct a probabilistic model of a specific bidder we constrain the sample using the corresponding lane descriptor. The variable $?bidder$ in the bidder lane descriptor, i.e. $Ann(l_B, Q_B)$ where $Q_B = ().\{?a : bidder ?bidder\}$, is bind to the individual of interest, e.g. $\{?bidder : \langle people/tom \rangle\}$. Given that $?bidder$ is used in a bridge descriptor we can merely recover those process instances $I_W = \langle \pi_i, \bar{v}_i \rangle$ where $\{?bidder : \langle people/tom \rangle\} \in \pi_i$. The corresponding \bar{v}_i are used for building another joint probabilistic distribution.

For example, assuming that Tom is a bidder that usually does not pay the items he wins we would have a distribution like the one shown in Table 3. Note that Tom might appear in other auctions as a seller but those instances are not retrieved.

Table 3: Joint probability distribution for $\{?bidder : \langle people/Tom \rangle\}$ in $\mathcal{M}_{Auction}$.

Cases	AO	BID	PAY	GS	TRA	REV_B	REV_S	AP	Comment
2	TRUE	?bidder	FALSE	FALSE				FAILURE	Bidder doesn't pay item
3	TRUE	?bidder	FALSE	FALSE			NEG	FAILURE	Bidder doesn't pay item. Seller rates negatively.
2	TRUE	?bidder	TRUE	TRUE	SELLER	POS	POS	SUCCESS	Item sold. Rating POS-POS.
1	TRUE	?bidder	TRUE	TRUE	SELLER	POS	NEG	SUCCESS	Item sold. Rating POS-NEG.

Using these 8 cases we apply parametric learning over the enriched and the compact model in order to compare the prediction on the auction outcome and the seller's review. As can be seen in Table 4, the probability of a successful outcome or a seller's positive review when Tom participates is lower than the average using both models. The difference is even clearer comparing the probability of payment ($V_{PAY} = TRUE$), which is the effect of an action attributed only to the bidder. This information would be used for deciding to ban a user from the on-line auction community, or at least forbidding awarding him in any further auction.

Table 4: Comparing predictions for a given bidder in $\mathcal{M}_{Auction}$.

	Enriched Model		Compact Model	
	?bidder	$\langle people/tom \rangle$?bidder	$\langle people/tom \rangle$
A posteriori probability				
$P(AP = SUCCESS BID = ?bidder)$	0.9142	0.3924	0.8715	0.3912
$P(REV_S = POS BID = ?bidder)$	0.8431	0.3091	0.8614	0.2939
$P(PAY = TRUE BID = ?bidder)$	0.9153	0.3888	0.9153	0.3888

4 Discussion

Following Judea Pearl’s causal approach [14], relations found by the Inferred Causation (IC) algorithm reflect direct or indirect causal relationships between cause and effect variables. Precedence between BPD nodes mapped to random variables in the BN were used for choosing the correct direction of undirected arcs found by the IC algorithm. In this way, temporal precedence encoded in the BPD is used for supporting causality on discovered conditional dependencies.

As shown in the case study, the IC algorithm complemented the conditional dependence graph obtained from the process workflow with additional causal relationships. This permitted to forecast the outcome of the process with 100% precision, and evaluating a new case based on the history of one actor of the process (the bad-bidder example).

The integration of semantic annotations in BPDs normalized for its transformation to Bayesian Networks has additional benefits. On one hand, it produces a hybrid representation of process instances where its probabilistic component is used for Bayesian learning, and its semantic component is used for filtering process instances.

Furthermore, discovering causal relations in other process scenarios would require: 1) normalizing a BPD representing the process, 2) annotating semantically event nodes using well-established vocabularies, 3) observing past process instances from public or proprietary Linked Data, and 4) applying the IC algorithm. Nevertheless, we identify three obstacles for our approach: BPDs must be normalized for generating a Bayesian Network from them, semantic annotation of BPDs requires human experts, and Linked Data that could be used for learning causal relationships must be available and be properly linked.

Finally, we would like to point out that despite stochastic BPD approaches supported by Markov Chain models [2, 3] can estimate the probability of observing certain event or condition at a time t , using a Bayesian Network we were able to discover causal relationships between events associated to non-contiguous events. These relations would permit to generate more accurate artificial process instances.

5 Conclusions

We integrated and extended semantic annotation and probabilistic inference on BPDs. As a result, a hybrid representation of past process instances was introduced. On one hand, its probabilistic component contains information for the Bayesian Network generated from normalized BPDs. On the other, the semantic component has information that permits selecting process instances with a given feature. In our case study, this characteristic is used for generating a probabilistic distribution of a given bidder from a on-line auction knowledge base.

The Pearl’s Inferred Causation (IC) algorithm was used for discovering causal relationships which could be selectively incorporated in the original graph. The resulting independence map (I-Map), with its corresponding probabilistic distribution, was used for deciding whether to award or not a bidder in an on-line auction scenario. Bidder’s past auctions were identified and extracted from the process’ joint probabilistic distribution by using the semantic component of process instances.

5.1 Future Work

A limitation of the probabilistic BPD normal form [4] is the lack of support for loops. Nevertheless, a loop-removal procedure can be implemented on BPDs using subprocesses and recursion. For this, the loop is encapsulated in a subprocess which is recursively called whenever the

loop occurs. The subprocess, specified through another BPD, can in turn be specified using the probabilistic normal form. The Bayesian Network generated for the subprocess would be trained as well through the observation of pass process instances. Further research must be done on the consequences of learning BNs from cycles.

On the other hand, semantic annotations of normalized BPDs can be further exploited for learning process traces from a knowledge base \mathcal{K} . This procedure must produce hybrid process instances where the probabilistic component must be aligned to the Bayesian Network obtained from the normalized BPD, whereas the semantic component would contain identifiers (URIs) of individuals described in \mathcal{K} .

Additionally, the unique BPD pool can be annotated as well, indicating the context on which the activity develops. For instance, it could be used for indicating whether the bidder knows the seller (`?bidder foaf:knows ?seller`). Pool annotation in this case, could be used for constraining the cases to use for building the CPD of the underlying Bayesian Network.

Finally, probabilistic models generated for each agent (lane) can in turn be used for assisting human activities through intelligent agents [15, 16].

6 Acknowledgments

This research was supported by Tecnológico de Monterrey through the Intelligent Systems research group, and by CONACyT through the grant CB-2011-01-167460.

References

- [1] OMG, *Business Process Model and Notation (BPMN), Version 2.0* (January 2011). URL <http://www.omg.org/spec/BPMN/2.0>
- [2] D. Prandi, P. Quaglia, N. Zannone, Formal analysis of bpmn via a translation into cows, in: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 5052 LNCS, 2008, pp. 249–263.
- [3] L. Herbert, R. Sharp, Precise Quantitative Analysis of Probabilistic Business Process Model and Notation Workflows, *Journal of Computing and Information Science in Engineering* 13 (1) (2013) 011007(1–9).
- [4] H. G. Ceballos, V. Flores-Solorio, J. P. Garcia, A Probabilistic BPMN Normal Form to Model and Advise Human Activities, in: M. Baldoni, L. Baresi, M. Dastani (Eds.), *Engineering Multi-Agent Systems: Third International Workshop, EMAS 2015, Istanbul, Turkey, LNAI 9318*, 2015.
- [5] H. Ceballos, E. Fernandez, J. Espinoza, Refining semantically annotated business process diagrams, in: *4th International Semantic Web and Linked Open Data Workshop (ISW-LOD 2016)*, Vol. 1807, *CEUR Workshop Proceedings*, 2016, pp. 11–21.
- [6] F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, P. F. Patel-Schneider (Eds.), *The Description Logic Handbook: Theory, Implementation, and Applications*, Cambridge University Press, New York, NY, USA, 2003.
- [7] E. Prud'hommeaux, A. Seaborne, SPARQL query language for RDF. W3C working draft., <http://www.w3.org/TR/2006/WD-rdf-sparql-query-20061004/> (October 2006).
- [8] T. Kuster, M. Lutzenberger, A. Hessler, B. Hirsh, Integrating process modelling into multi-agent systems engineering, in: *Multiagent and Grid Systems*, 2012, pp. 105–124.
- [9] S. Muggleton, *Advances in Inductive Logic Programming*, IOS Press, 1996, Ch. Stochastic Logic Programs, pp. 254–264.
- [10] M. Richardson, P. Domingos, Markov logic networks, *Machine Learning* 62 (2006) 107–136. doi: [10.1007/s10994-006-5833-1](https://doi.org/10.1007/s10994-006-5833-1).

- [11] K. B. Laskey, MEBN: A language for first-order bayesian knowledge bases, *Artificial Intelligence* 172 (2008) 140–178.
- [12] J. Pearl, *Causality. Models, Reasoning, and Inference*, Cambridge University Press, 2000.
- [13] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I. H. Witten, *The weka data mining software: An update*, *SIGKDD Explorations* 11 (1).
URL <http://www.kdd.org/explorations/issues/11-1-2009-07/p2V11n1.pdf>
- [14] J. Pearl, *Causality. Models, Reasoning, and Inference*, Cambridge University Press, 2000.
- [15] H. Ceballos, J. P. Garcia-Vazquez, R. Brena, Using activity theory and causal diagrams for designing multiagent systems that assist human activities, in: F. Castro, A. Gelbukh, M. Gonzalez (Eds.), *Advances in Artificial Intelligence and Its Applications*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013, pp. 185–198.
- [16] H. Ceballos, F. Cantu, Modelling intelligent agents through causality theory, in: *2007 Sixth Mexican International Conference on Artificial Intelligence, Special Session (MICAI)*, IEEE, 2008, pp. 201–210.