Probabilistic Logics for Declarative Process Mining

Elena Bellodi

Department of Engineering
University of Ferrara, Italy

LPNMR 2026, September 8th



DE: Department of Engineering Ferrara

Who I am

- Artificial Intelligence@UniFE (AI@UniFE) research group (`https://ai.unife.it/`)
- Coordinator of the Statistical Relational AI Laboratory (`https://ai.unife.it/starai-lab`)
- Artificial Intelligence, Machine Learning, Probabilistic Logic Programming, Description Logics and Semantic Web, Process Mining, Smart City Data Analysis

Acknowledgments

- This research was funded by the Italian Ministerial grant PRIN 2022 *"Probabilistic Declarative Process Mining (PRODE)"*, n. 20224C9HXA – CUP F53D23004240006, funded by the European Union – Next Generation EU
- University of Bologna & University of Calabria
- Part of the material presented in these slides was developed by **Michela Vespa**, a Ph.D. student under my supervision, enrolled in the *Italian National Ph.D. Program in Artificial Intelligence - Industry* and affiliated with the Department of Engineering, University of Ferrara, Italy







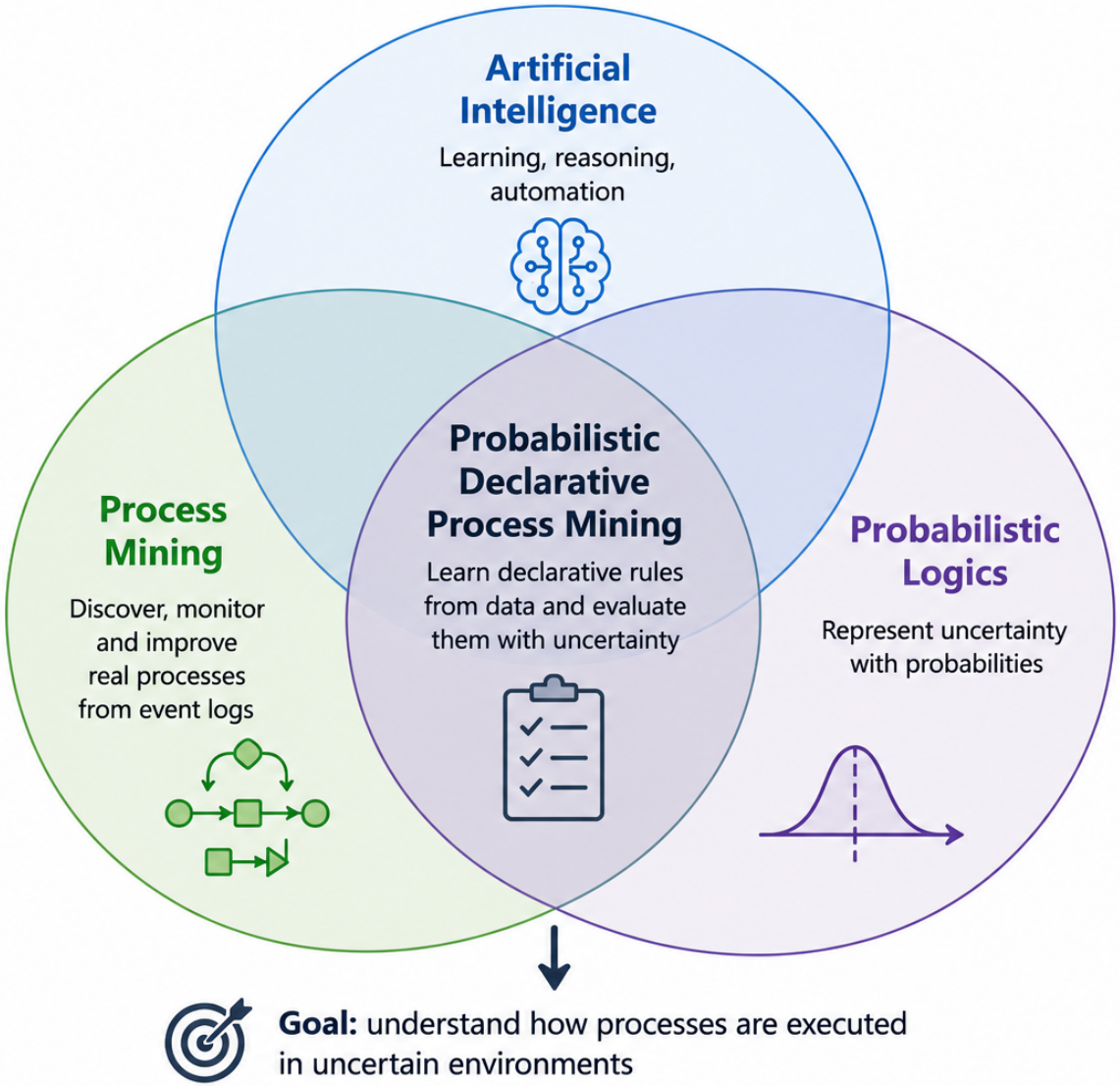




Outline

1. What is a Process?
2. What is Process Mining?
3. Declarative Process Mining
4. Why Boolean logic is not enough
5. Probabilistic Declarative Process Mining
6. Future Work

Graphical Outline



Outline

1 What is a Process?

2 What is Process Mining?

3 Declarative Process Mining

4 Why Boolean logic is not enough

5 Probabilistic Declarative Process Mining

6 Future Work

What is a Process?

- A Process is a set of events/activities or tasks that are performed in a specific sequence to achieve a (business) goal.
- Events happen inside a machine, an enterprise information system, a hospital, a social network, a transportation system,...
- The spectacular growth of Big Data and Big Data Analytics techniques made it possible to record, derive and analyze events.



Courtesy of https://7hype.com/blog/big-data/

A Process inside a Hospital

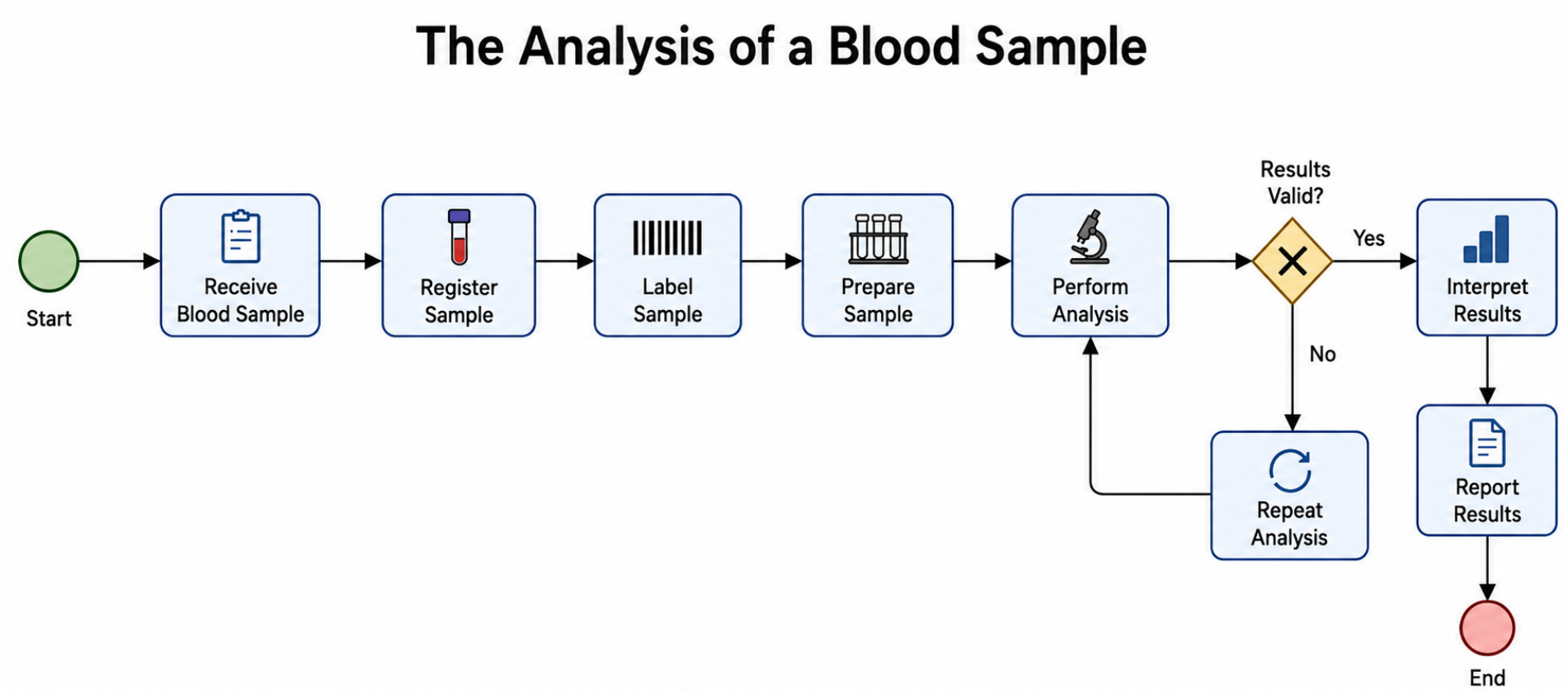


Image generated with chatGPT, accessed on 2026/08/17.

A (simple) E-commerce Process

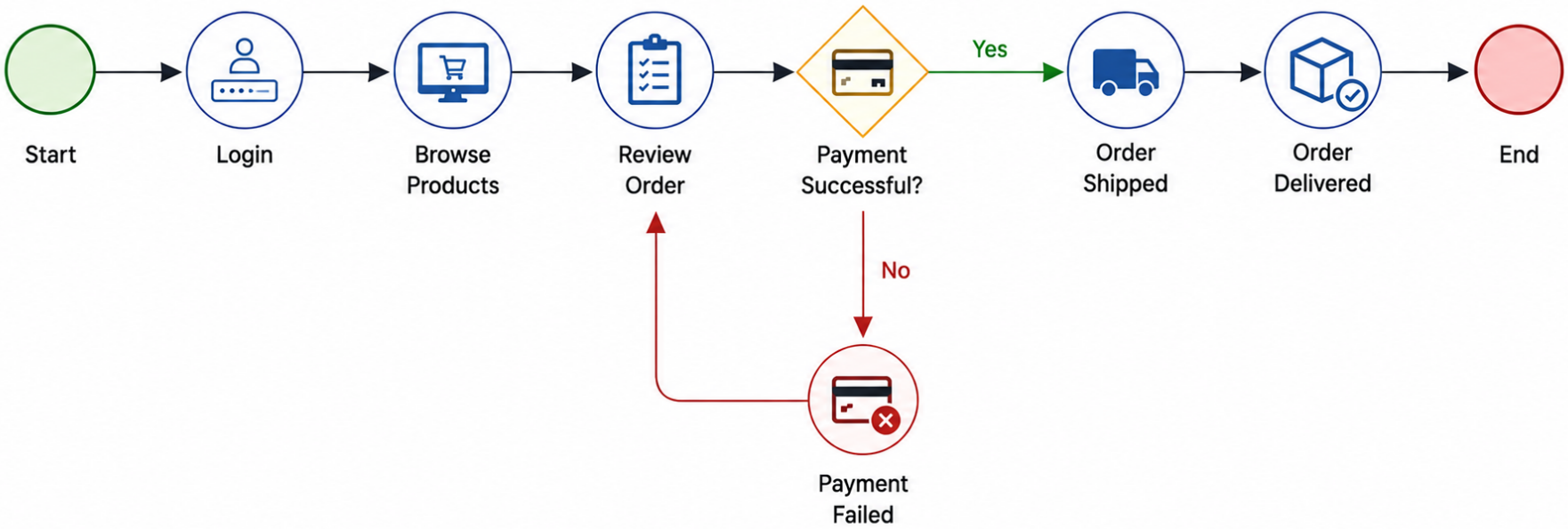


Image generated with chatGPT, accessed on 2026/08/17.

Outline

1 What is a Process?

2 What is Process Mining?

3 Declarative Process Mining

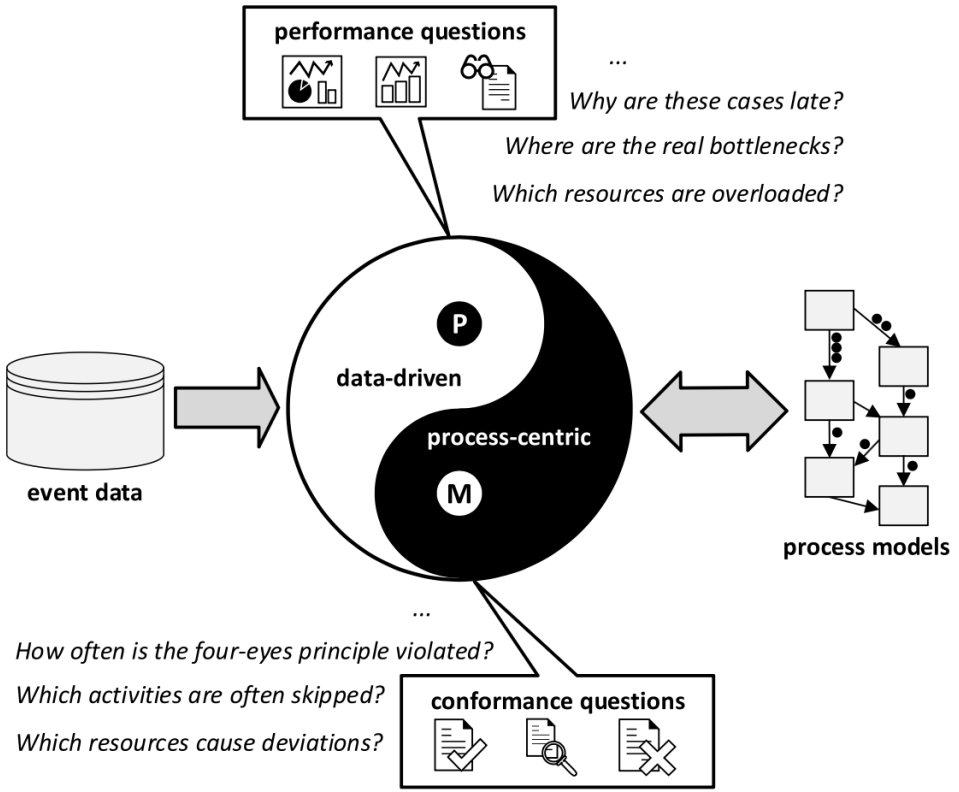
4 Why Boolean logic is not enough

5 Probabilistic Declarative Process Mining

6 Future Work

What is Process Mining?

- The idea of **Process Mining (PM)** is to monitor, discover and improve real processes by extracting knowledge from event logs readily available in today's (information) systems

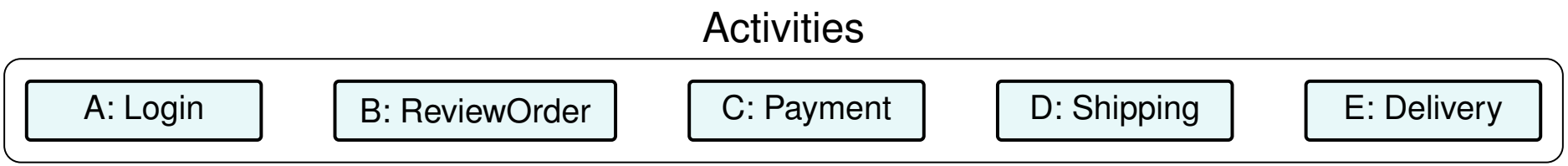


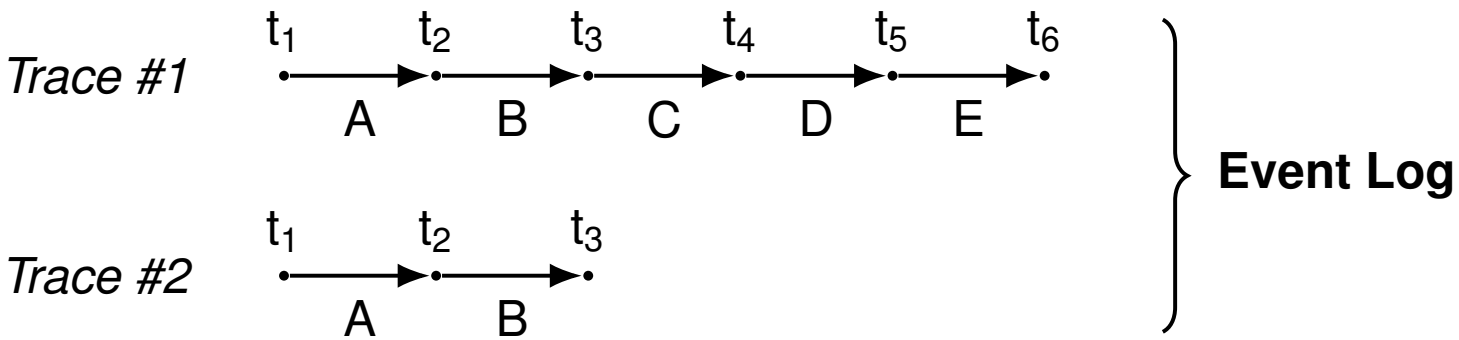
Courtesy of [van der Aalst, 2016], page 26.

Basic Definitions

- A **Trace** is a process execution (a process instance)
- **Activities/Events** are identified by names and typically marked with a temporal timestamp t
- Traces are represented as sequences of activity names, ordered by their timestamps
- A **Log** is a finite set of traces

Basic Definitions



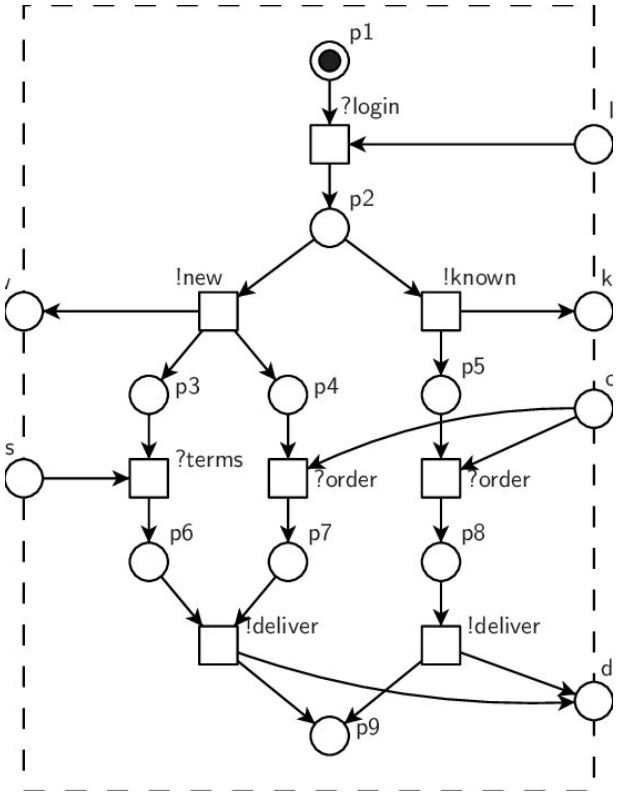


Basic Definitions

- A **process model** is a visual and formal representation of a business process, produced by applying Data Mining algorithms to event logs or manually designed by the user.

Basic Definitions

- **Procedural**: Specifies the sequence of activities to be performed with a rigid and explicit control flow (Petri Nets, Event-Driven Process Chains, BPMN, ...)

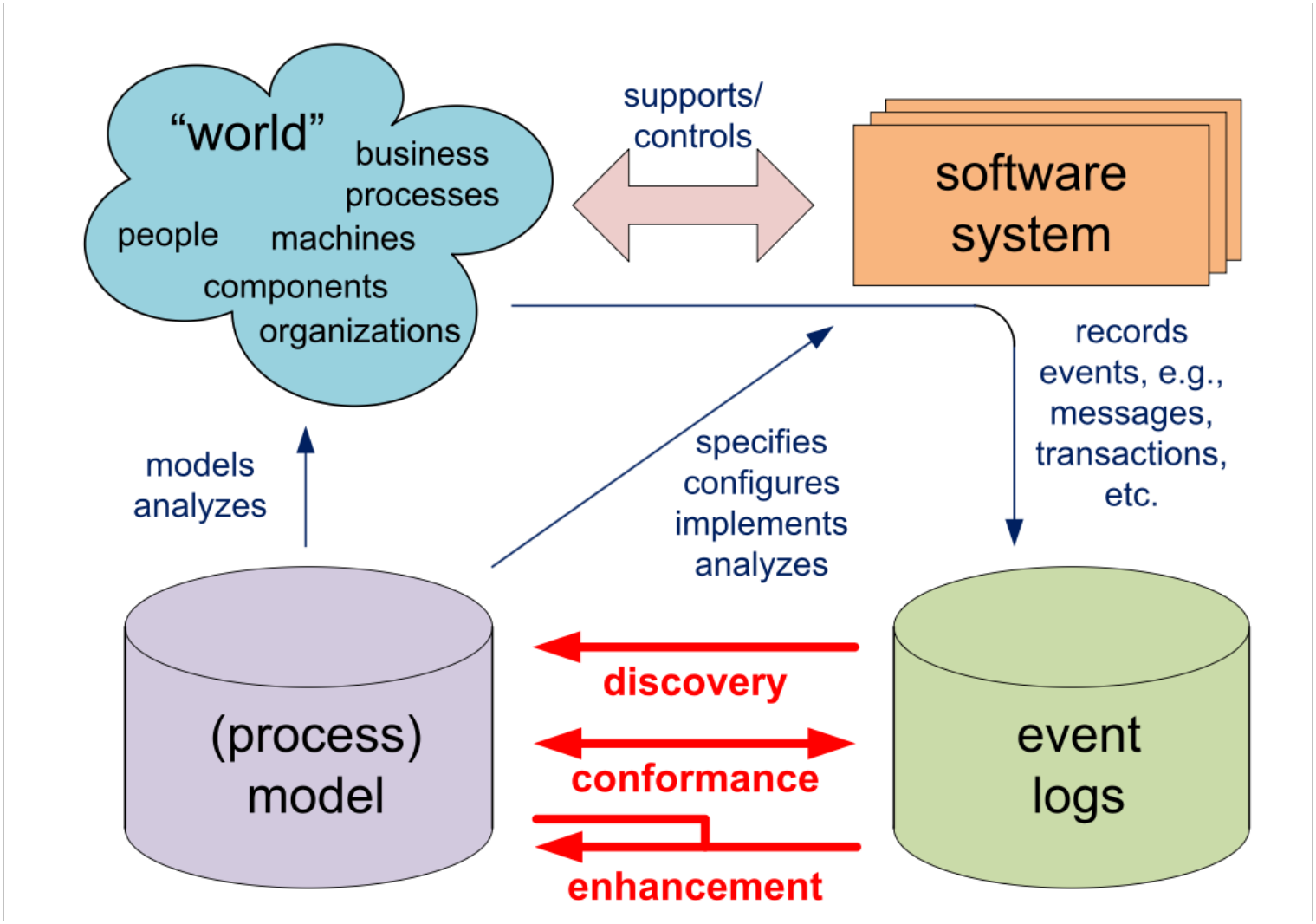


Courtesy of "Letia, I.A. and Goron A., Using arguments on requirements for services. 2011 IEEE 7th International Conference on Intelligent Computer Communication and Processing. IEEE."

Basic Definitions

- **Declarative**: Specifies what must/should hold (not how) by means of constraints (Declare)
- Example (Declare):
  - Precedence(A,B): if B occurs, then A must have occurred before
  - Response(A,B): if A occurs, then B must occur at some point in the future
  - ```
    Precedence(Login, ReviewOrder)
    Precedence(ReviewOrder, Payment)
    Precedence(Payment_succ, Shipping)
    Response(Payment_failed, ReviewOrder)
    Precedence(Shipping, Delivery)
    ```
    ```

Main Tasks



Courtesy of [van der Aalst et al., 2012], page 6.

Main Tasks [van der Aalst, 2016]

- **Conformance checking** tries to compare the behavior of a process model and the behavior recorded in an event log to find commonalities and discrepancies.
- A **process discovery** algorithm is a function that maps a log onto a process model such that the model is *representative* for the behavior seen in the event log.
- **Enhancement** aims at improving an existing process model using information about the actual process recorded in an event log.

Outline

1 What is a Process?

2 What is Process Mining?

3 Declarative Process Mining

4 Why Boolean logic is not enough

5 Probabilistic Declarative Process Mining

6 Future Work

The Declare Language

- *"Everything is possible unless explicitly forbidden"*
- Based on constraint templates for different relation types
- Each constraint template has (1) an unique *name*, (2) a *semantics* specified in Linear Temporal Logic (LTL) and (3) *graphical representation*
- It is possible to learn Declare models and the graphical language is also suitable for conformance checking (given an event log, by checking the constraints)

The Declare Language

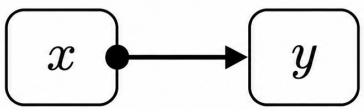
- In the original formulation [Pesic et al., 2007] the semantics was given using the Linear Temporal Logic or LTL [Clarke et al., 2001]
- LTL is a special type of logic that uses (in addition to classical logical operators $\neg, \vee, \wedge$) several *temporal or modal* operators: always ($\Box$), eventually ($\Diamond$), until ($\mathcal{U}$), next time ($\bigcirc$)
- [De Giacomo and Vardi, 2013] introduced the LTL_f logic, i.e. LTL over **finite traces**: same syntax, semantics in terms of interpretations over *finite* sequences of consecutive instants of time
- LTL_f has exactly the same expressive power of FOL

The Declare Language

- Given a propositional symbol p:

$$\phi := p|\neg\phi|(\phi \vee \phi)|(\phi \wedge \phi)|\bigcirc\phi|\Diamond\phi|\Box\phi|(\phi \mathcal{U} \phi)$$

- The Response(A,B) constraint corresponds to the LTL expression $\Box(A \Rightarrow \Diamond(B))$
- The language allows one to work with a graphical representation of templates, while the underlying LTL_f formula remains hidden. For instance:



$Response(x, y)$

If x occurs, then y occurs after x

Outline

1 What is a Process?

2 What is Process Mining?

3 Declarative Process Mining

4 Why Boolean logic is not enough

5 Probabilistic Declarative Process Mining

6 Future Work

Why Boolean logic is not enough

Until recently everything was Boolean.

- **Declarative process models** were purely logical: (Declare) constraints are crisp (certain) and they are either satisfied or violated by a log.

  $\Rightarrow$ What if I have to express both soft and mandatory constraints, such as recommended *vs* hard clinical guidelines?

- **Events** were supposed to happen with 100% certainty

  $\Rightarrow$ What if I have malfunctioning tools, incomplete information, erroneous manually-entered data?

Outline

1 What is a Process?

2 What is Process Mining?

3 Declarative Process Mining

4 Why Boolean logic is not enough

5 Probabilistic Declarative Process Mining

6 Future Work

Probabilistic Declarative Process Mining

- **ProbDeclare** (*Alman et al. 2022*)
- The first framework which introduces the notion of *probabilistic process constraints* and tackles discovery, conformance checking and monitoring with existing data mining algorithms for PM
- A constraint with assigned probability and relational operator $(0.9, \geq)$ is satisfied if the number of traces satisfying it is at least $(\geq)$ the 90% of the total traces
- Semantics as relative frequency of traces satisfying a constraint

Probabilistic Declarative Process Mining

- May we do something **more**, by following some different approach and semantics?
- Our research idea is based on the Distribution Semantics borrowed from Probabilistic Logic Programming (PLP).
- We show how to model both probabilistic events and probabilistic process models in a declarative setting.
- We show how to perform (up to now) conformance checking and discovery

Probabilistic Logic Programming

- Logic is useful to model domains with complex relationships among entities
- Uncertainty in real-world domains is ubiquitous and can be modeled by means of probability theory
- Probabilistic Logic Programming (PLP) combines the use of LP and probability to model both structured and uncertain domains
- Approaches in this direction can be broadly classified into two categories: those based on the Distribution Semantics [Sato, 1995] and those that follow a Knowledge Base Model Construction (KBMC) approach

LPAD

- Logic Programs with Annotated Disjunctions (LPAD) [Vennekens et al., 2004]
- An annotated disjunctive clause C is written as:

$$h_1 : p_1; \ldots; h_n : p_n \leftarrow b_1, \ldots, b_r.$$

- $\{p_1, \ldots, p_n\}$ are probabilities in $[0, 1]$ such that $\sum_{i=1}^{n} p_i \leq 1$
- A simple LPAD

$$\begin{aligned} &sneezing(X) : 0.7 \leftarrow flu(X). \\ &sneezing(X) : 0.8 \leftarrow hay_fever(X). \\ &flu(bob). \\ &hay_fever(bob). \end{aligned}$$

Distribution Semantics

- A Probabilistic Logic Program under the Distribution Semantics defines a probability distribution over normal logic programs (called instances/possible worlds/**worlds**)
- Finite Herbrand universe, finite set of groundings of each clause/fact
- Atomic choice (C, θ, i): selection of the i-th head atom for the grounding $C\theta$ of clause C
- Example: $C_1 = sneezing(X) : 0.7; null : 0.3 \leftarrow flu(X).$
- The atomic choice $(C_1, \{X/bob\}, 1)$ selects $sneezing(bob)$ with $\theta = \{X/bob\}$

Distribution Semantics

- Selection σ: a total set of atomic choices (one atomic choice for every grounding of each clause of the LPAD)
- **A selection σ identifies a normal logic program w_σ called world**
- The probability of w_σ is $P(w_\sigma) = P(\sigma) = \prod_{(C,\theta,i)\in\sigma} p_{(C,i)}$
- $P(w)$ distribution over a finite set of worlds: $\sum_w P(w) = 1$

Distribution Semantics

Possible worlds for the previous LPAD:

$sneezing(bob) \leftarrow flu(bob).$
$sneezing(bob) \leftarrow hay_fever(bob).$
$flu(bob).$
$hay_fever(bob).$
$\mathbf{P(w_1) = 0.7 \times 0.8 = 0.56}$

$null \leftarrow flu(bob).$
$sneezing(bob) \leftarrow hay_fever(bob).$
$flu(bob).$
$hay_fever(bob).$
$\mathbf{P(w_2) = 0.3 \times 0.8 = 0.24}$

$sneezing(bob) \leftarrow flu(bob).$
$null \leftarrow hay_fever(bob).$
$flu(bob).$
$hay_fever(bob).$
$\mathbf{P(w_3) = 0.7 \times 0.2 = 0.14}$

$null \leftarrow flu(bob).$
$null \leftarrow hay_fever(bob).$
$flu(bob).$
$hay_fever(bob).$
$\mathbf{P(w_4) = 0.3 \times 0.2 = 0.06}$

Distribution Semantics

- The probability of a query Q given a world w is $P(Q|w) = 1$ if $w \models Q$ and 0 otherwise.

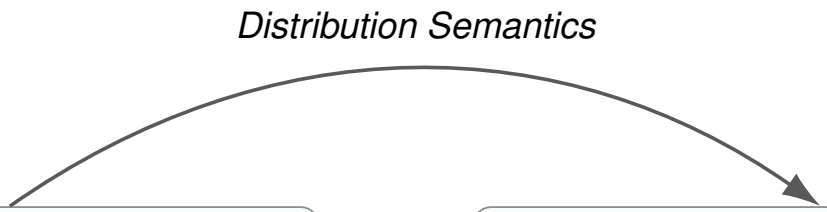
$$P(Q) = \sum_{w \in W} P(Q, w) = \sum_{w \in W} P(Q|w)P(w) = \sum_{w \in W : w \models Q} P(w)$$

- $Q = sneezing(bob)$ is true in the first 3 worlds
- $P(Q) = 0.7 \times 0.8 + 0.3 \times 0.8 + 0.7 \times 0.2 = 0.94$

Distribution Semantics for PM

- The Distribution Semantics from PLP can provide a formal basis for a *unifying* modelling framework (events, constraints, ...) for Probabilistic Declarative Process Mining

Putting It All Together



Declarative PM

Log

$\langle A, B, C \rangle$

$\langle A, D \rangle$

$\langle E, F \rangle$

Declare Process Model

init(A)

response(A,B)

absence(C)

Probabilistic Declarative PM

Log

$\langle A, B, C \rangle$

$\langle A,\ 0.6 :: B,\ D \rangle$

$\langle E, F \rangle$

Declare Process Model

0.9 :: init(A)

0.8 :: response(A,B)

absence(C)

Probabilistic Events and Traces

Probabilistic Event [Vespa et al., 2024b]

A Probabilistic Event is a couple **p** :: **E** where

- $E \in \mathcal{A}$ is an event and $\mathcal{A}$ is the set of activity names.
- $p \in [0, 1]$ is **a degree of confidence in its occurrence**. If $p = 1$ the event is certain.

Probabilistic Trace [Vespa et al., 2024b]

A Probabilistic Trace is a trace where at least one event is probabilistic.

NB: p is user-defined.

How can we use them in practice?

- In a hospital some events may have occurred but **not logged**.

$$\langle\ 0.7 :: \mathit{check_fever}(\mathit{sam}, 1),\ \mathit{drug_prescription}(\mathit{sam}, 2),\ \mathit{check_fever}(\mathit{sam}, 3),\ 0.95 :: \mathit{check_fever}(\mathit{sam}, 4),\ \mathit{discharge}(\mathit{sam}, 5)\ \rangle$$

- Two *check_fever* events likely occurred for patient *sam*, with a degree of confidence, given the patient's treatment and the knowledge of the standard protocol, so we recorded them in the trace.

Probabilistic Constraints

Probabilistic Constraint [Vespa et al., 2024a]

Let REP be a finite, non-empty set of constraint templates and $\mathcal{A} = a_1, \ldots, a_m$ be a set of activity names.
We denote a generic i-th probabilistic constraint as:

$$p_i :: \mathrm{C}_i(a_1, \ldots, a_m)$$

where $p_i \in [0..1]$ represents the strength or importance of the constraint $\mathrm{C} \in$ REP instantiated over $\mathcal{A}$.

- A higher value of p indicates a stronger or more important constraint.

Probabilistic Declare Models

Probabilistic Declarative Process Specification [Vespa et al., 2024a]

A Probabilistic Declarative Process Specification PDS is a Declarative Process Specification DS where at least one constraint $c_i \in DS$ is a probabilistic constraint.

How can we use them in practice?

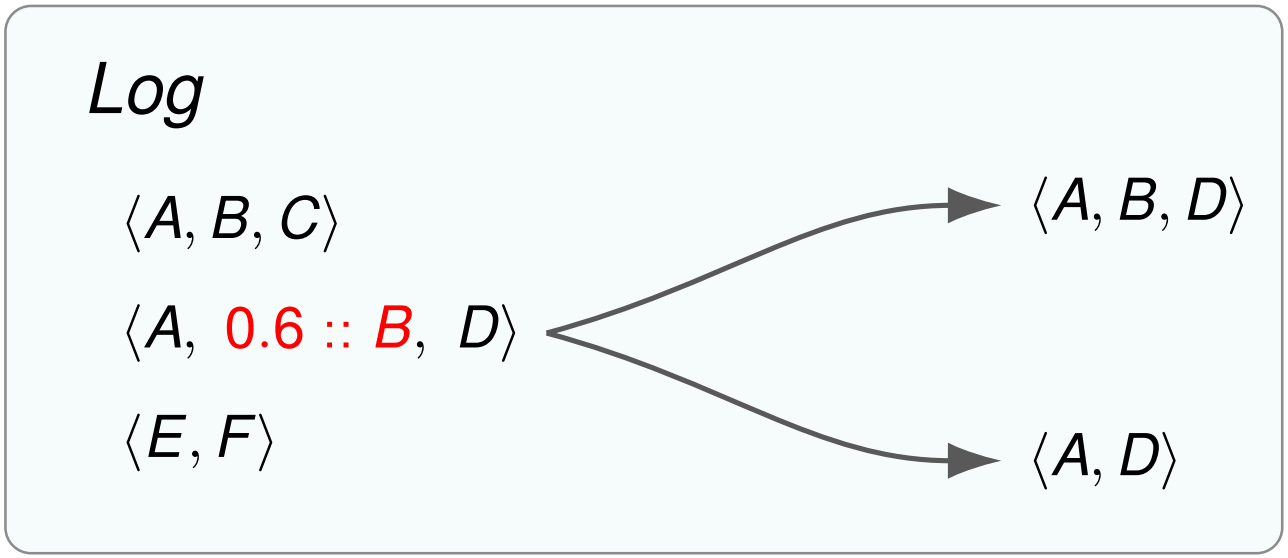
- In a hospital some guidelines are recommended (**soft constraints**) and others are mandatory (**hard constraints**).
- Given the PDS (with P patient):

$$\{ \quad c_1 = \mathit{Precedence}(\mathit{preanesthesia}(P), \mathit{start_surgery}(P)) \quad c_2 = 0.9 :: \mathit{Response}(\mathit{end_surgery}(P), \mathit{audit}(P)) \quad \}$$

we are expressing that:

- *Preanesthesia* must precede *surgery* (hard)
- *Audit* is highly recommended but not mandatory (soft)

The Distribution Semantics for Probabilistic Traces



$P(w_{\sigma_1}) = 0.6$

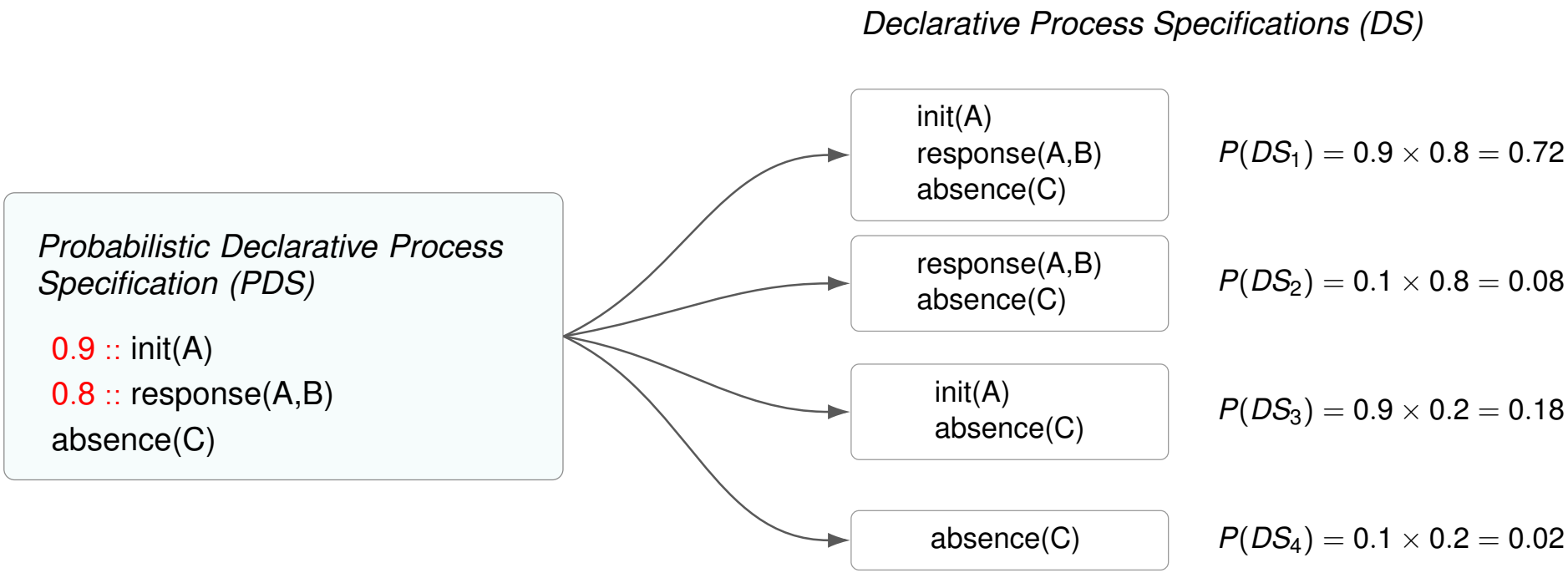
$P(w_{\sigma_2}) = 0.4$

- w_{σ_1} and w_{σ_2} are possible worlds according to the distribution semantics

How did we get there?

- Atomic Choice (E, k), with $k \in \{0, 1\}$: $k = 1$ ($k = 0$) includes (excludes) event $p :: E$ with probability p $(1 - p)$ in (from) a selection
- Selection σ: one atomic choice for every probabilistic event in a trace t
- **A selection identifies a non-probabilistic (regular) trace (a possible world)** $w_\sigma(t)$
- The probability of $w_\sigma(t)$ is $P(w_\sigma(t)) = P(\sigma) = \prod_{(E_i,1)\in\sigma(t)} p_i \prod_{(E_i,0)\in\sigma(t)} (1 - p_i)$

The Distribution Semantics for Probabilistic Declarative Process Specifications



- $\sum_i P(DS_i) = 1$

How did we get there?

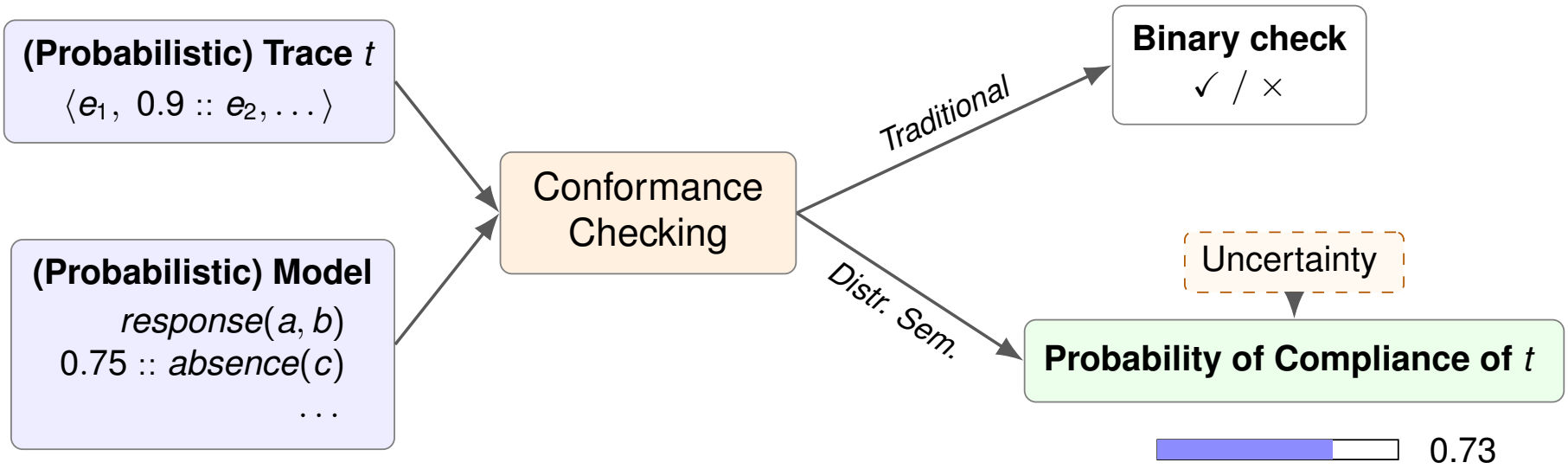
- Atomic Choice (c_i, k), with $k \in \{0, 1\}$: $k = 1$ ($k = 0$) includes (excludes) constraint $p_i :: c_i$ with probability p_i $(1 - p_i)$ in (from) a PDS
- Selection σ: one atomic choice for every probabilistic constraint
- **A selection identifies a declarative specification DS (a possible world)** w_σ
- The probability of the DS is $P(DS) = P(w_\sigma) = P(\sigma) = \prod_{(c_i,1)\in\sigma} p_i \prod_{(c_i,0)\in\sigma} (1 - p_i)$

Probabilistic Declarative PM Tasks

How to address the tasks of **conformance checking** and **discovery** for Declarative Process Mining under the Distribution Semantics?

Declarative Conformance Checking

Evaluating if a trace or a log is compliant with a declarative process specification (process model).



Conformance Checking

1) Traditional/Binary Approach

Compliance/violation of a trace versus a Constraint

A trace t is *compliant* with a Declare constraint if, named φ the corresponding logic formula modelling that constraint, it holds $t \models \varphi$.
A trace t *violates* a Declare constraint if it does not entail the corresponding formula φ, i.e. if $t \not\models \varphi$.

Compliance/violation of a trace versus a Model

A trace is *compliant* w.r.t. a declarative process specification DS iff it satisfies *all* its formulas φ.

Probabilistic Declarative Conformance Checking

2) ProbDeclare:

- A probabilistic constraint is satisfied over the log if:
  The number of traces satisfying the constraint over the log cardinality achieves the mathematical relation established by the relational operator and the probability assigned.

- If the constraint is $\langle \varphi, \geq, 0.8 \rangle$, it means that:
  At least 80% of the traces in the log must satisfy the condition φ.

Probabilistic Declarative Conformance Checking

3) Ours: we return the probability of compliance, a value $\in$ [0,1] indicating "how much" a trace is compliant to a PDS.

1. Compliance of a regular trace versus a PDS
2. Compliance of a probabilistic trace versus a PDS

Compliance of a regular trace versus a PDS

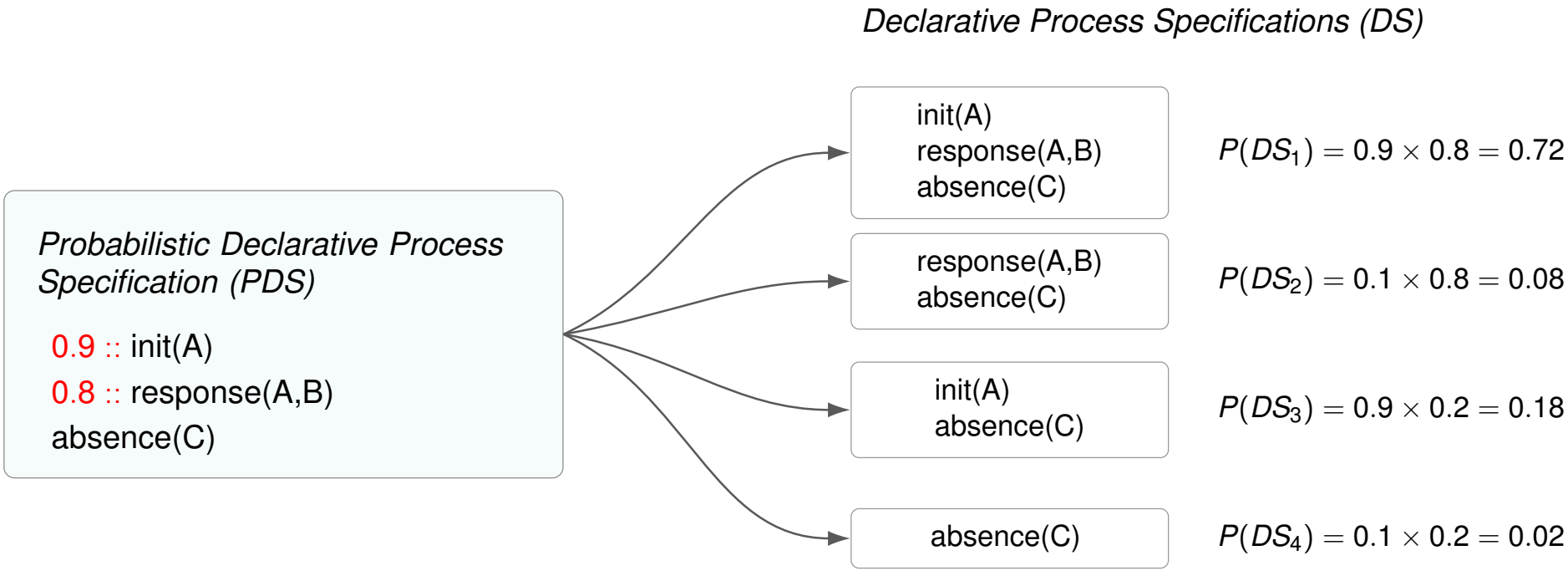
- Given a trace t and a PDS, the compliance $Comp$ of t vs the PDS is:

$$Comp(t, PDS) = \sum_{t \models DS_i} P(DS_i)$$

  where DS_i are deterministic specifications generated from the PDS.
- Sum of the probabilities of all the process models where the trace is compliant.
- The number of possible worlds DS_i grows exponentially with the number of probabilistic constraints: for n constraints, 2^n models

Compliance of a regular trace versus a PDS

- Check the compliance of the trace $t = \langle A, D \rangle$ w.r.t all the DS.



- $Comp(t, PDS) = 0.18 + 0.02 = 0.2$.

Compliance of a regular trace versus a PDS

- It is possible to avoid the full enumeration of worlds
- In [Vespa and Bellodi, 2025] we show that the probability of compliance depends only on the number of groundings m_i of the constraints $c_i \in PDS$ not satisfied in the trace t:

$$Comp(t, PDS) = \prod_{i=1}^{n} (1 - p_i)^{m_i}$$

where $p_i \in [0, 1]$ is the probability that constraint C_i is included in the PDS.

- **Complexity:** $O(n \log m)$, where n is the number of constraints and m the max number of groundings

Compliance of a probabilistic trace versus a PDS

Compliance of a probabilistic trace versus a PDS

Let $w_{\sigma_i}(t)$ be regular traces generated from a probabilistic trace t, and DS_j declarative specifications from a PDS. Then the compliance of t versus the PDS is:

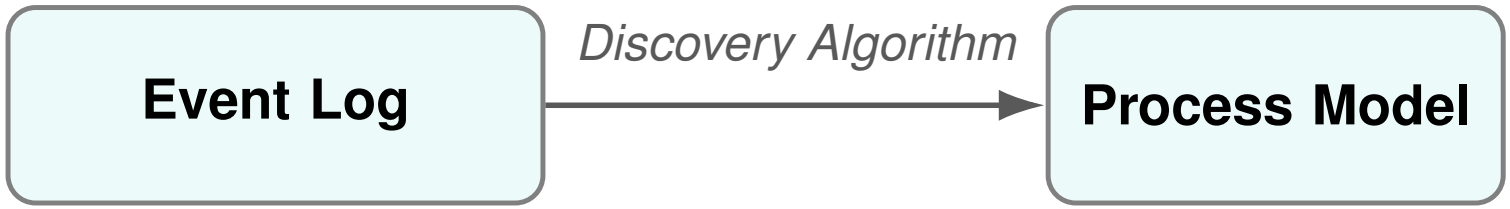
$$Comp(t, PDS) = \sum_{\substack{w_{\sigma_i}(t) \\ DS_j}} \begin{cases} P(w_{\sigma_i}(t)) \cdot P(DS_j) & \text{if } w_{\sigma_i}(t) \textit{ is compliant with } DS_j, \\ 0 & \text{otherwise.} \end{cases}$$

- Same problem: the number of (non-probabilistic) traces $w_i(t)$ generated from a probabilistic one grows exponentially: for n probabilistic events, 2^n regular traces

- We solved the exponential growth of the DS_i
- We have implemented *(under review)* several optimizations for this second problem
- We have provided *(under review)* a formula for computing the probability of compliance of *a set* of traces (log) versus a PDS

To be continued....

Probabilistic Process Discovery



t_1 : login(u_1,1),purchase(u_1,o_1,2),payment(u_1,o_1,s_1,3)
t_2 : login(u_2,1),purchase(u_2,o_2,2),payment(u_2,o_2,s_2,3),
shipping(o_2,4),delivery(o_2,5)
t_3 : login(u_3,1)

$p_1 :: c_1$
$p_2 :: c_2$
...

Probabilistic Process Discovery

- ProbDeclare applies traditional existing PM discovery algorithms then computes the relative frequency.
- We would like to learn both c_i and p_i, with p_i **meaning strength/importance** (according to our semantics), not relative frequency.
- We discover the probabilistic declarative specification from traces labeled compliant (+) and non compliant (-).
- Compliant/Non compliant traces are mapped onto +/- logical interpretations.

Probabilistic Discovery: Our Idea

1. Formulate probabilistic declarative process discovery as a problem of discriminative learning from interpretations, using Probabilistic Constraint Logic Theories (PCLTs)
2. Apply PASCAL (*ProbAbiliStic inductive Constraint Logic*) [Riguzzi et al., 2021], a supervised learning algorithm that learns PCLTs
3. Good results on two synthetic logs (e-commerce, medical): we recover most ground-truth constraints [Vespa et al., 2026]

Probabilistic Constraint Logic Theories (PCLT)

- A PCLT is a set of **probabilistic Integrity Constraints (PICs)**, each annotated with a probability p_i representing their **importance/strength**

$$p_i :: L_1, \ldots, L_b \rightarrow \exists(P_1); \ldots; \exists(P_n); \forall\neg(N_1); \ldots; \forall\neg(N_m)$$

- PCLT can express the logic formulas of a PDS

$$0.85 :: response(purchase(Order), shipping(Order))$$

$$\downarrow$$

$$0.85 :: purchase(Order, T_1) \rightarrow shipping(Order, T_2) \wedge T_2 > T_1$$

- A PCLT can be seen as a Process Model as it assigns any trace a **probability of being compliant to the model**

Results (E-commerce Log)

GROUND TRUTH

access $\rightarrow$ login $\rightarrow$ purchase $\rightarrow$ shipping $\rightarrow$ feedback
access $\rightarrow$ login $\rightarrow$ purchase $\rightarrow$ feedback
access $\rightarrow$ login $\rightarrow$ purchase $\rightarrow$ shipping
access $\rightarrow$ login $\rightarrow$ purchase
access $\rightarrow$ purchase $\rightarrow$ shipping
access $\rightarrow$ purchase

LEARNED PCLT

$0.262 :: \text{access}(T_1) \rightarrow \text{purchase}(\text{I},\text{P},T_2) \wedge T_2 > T_1$

$0.255 :: \text{access}(T_1) \rightarrow \text{login}(\text{G},\text{A},T_2) \wedge T_2 > T_1$

$0.253 :: \text{purchase}(\text{I},\text{P},T_1) \rightarrow \text{feedback}(\text{R},T_2) \wedge T_2 > T_1$

$0.262 :: \text{login}(\text{G},\text{A},T_1) \rightarrow \text{feedback}(\text{R},T_2) \wedge T_2 > T_1$

TEST SET

AUC-ROC	**0.84**
AUC-PR	0.79
Log-likelihood	-663.1

WHAT IS CAPTURED

PICs (1), (3), (4) link activities not consecutive in time.

Outline

1 What is a Process?

2 What is Process Mining?

3 Declarative Process Mining

4 Why Boolean logic is not enough

5 Probabilistic Declarative Process Mining

6 Future Work

Future Work

- Making the conformance checking algorithm efficient with respect to the trace dimension (under review)
- Integration of uncertainty into event attributes, such as timestamps and resources
- Causal probabilistic discovery
- Process discovery with constraint selection
- Monitoring
- ...

Thank you for your attention

Questions?

References I


Clarke, E. M., Grumberg, O., and Peled, D. A. (2001).
Model checking, 1st Edition.
MIT Press.


De Giacomo, G. and Vardi, M. Y. (2013).
Linear temporal logic and linear dynamic logic on finite traces.
In *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, IJCAI '13, pages 854–860. AAAI Press.


Pesic, M., Schonenberg, H., and van der Aalst, W. M. P. (2007).
Declare: Full support for loosely-structured processes.
In *Proceedings of the 11th IEEE International Enterprise Distributed Object Computing Conference*, EDOC '07, page 287, USA. IEEE Computer Society.

References II



Riguzzi, F., Bellodi, E., Zese, R., Alberti, M., and Lamma, E. (2021).
Probabilistic inductive constraint logic.
Machine Learning, 110(4):723–754.



Sato, T. (1995).
A statistical learning method for logic programs with distribution semantics.
In *International Conference on Logic Programming*, pages 715–729. MIT Press.



van der Aalst, W. (2016).
Process Mining: Data Science in Action.
Springer, Berlin.

References III



van der Aalst, W., Adriansyah, A., de Medeiros, A. K. A., Arcieri, F., Baier, T., Blickle, T., Bose, J. C., van den Brand, P., Brandtjen, R., Buijs, J., Burattin, A., Carmona, J., Castellanos, M., Claes, J., Cook, J., Costantini, N., Curbera, F., Damiani, E., de Leoni, M., Delias, P., van Dongen, B. F., Dumas, M., Dustdar, S., Fahland, D., Ferreira, D. R., Gaaloul, W., van Geffen, F., Goel, S., Günther, C., Guzzo, A., Harmon, P., ter Hofstede, A., Hoogland, J., Ingvaldsen, J. E., Kato, K., Kuhn, R., Kumar, A., La Rosa, M., Maggi, F., Malerba, D., Mans, R. S., Manuel, A., McCreesh, M., Mello, P., Mendling, J., Montali, M., Motahari-Nezhad, H. R., zur Muehlen, M., Munoz-Gama, J., Pontieri, L., Ribeiro, J., Rozinat, A., Seguel Pérez, H., Seguel Pérez, R., Sepúlveda, M., Sinur, J., Soffer, P., Song, M., Sperduti, A., Stilo, G., Stoel, C., Swenson, K., Talamo, M., Tan, W., Turner, C., Vanthienen, J., Varvaressos, G.,

References IV

Verbeek, E., Verdonk, M., Vigo, R., Wang, J., Weber, B., Weidlich, M., Weijters, T., Wen, L., Westergaard, M., and Wynn, M. (2012).
Process mining manifesto.
In Daniel, F., Barkaoui, K., and Dustdar, S., editors, *Business Process Management Workshops*, pages 169–194, Berlin, Heidelberg. Springer Berlin Heidelberg.



Vennekens, J., Verbaeten, S., and Bruynooghe, M. (2004).
Logic programs with annotated disjunctions.
In *International Conference on Logic Programming*, volume 3131 of *LNCS*, pages 195–209. Springer.

References V



Vespa, M. and Bellodi, E. (2025).
A scalable approach to probabilistic compliance in declarative process mining.
In Azzolini, D., Batsakis, S., Santana, M. A. B., Bruno, P., Cecchi, L. A., Fandinno, J., Hecher, M., Maratea, M., Morales, J. F., Muniz, B., Papadakis, E., Robaldo, L., Serafini, L., Scudo, F. L., Tachmazidis, I., Tarzariol, A., Vallati, M., and Wyner, A., editors, *Joint Proceedings of the Workshops and Doctoral Consortium of the 41st International Conference on Logic Programming (ICLP-WS-DC 2025). Rende, Italy, September 9-13, 2025.*, volume 4117 of *CEUR Workshop Proceedings*, pages 1–12. CEUR-WS.org.

References VI



Vespa, M., Bellodi, E., Chesani, F., Loreti, D., Mello, P., Lamma, E., and Ciampolini, A. (2024a).
Probabilistic compliance in declarative process mining.
In Giacomo, G. D., Fionda, V., Fournier, F., Ielo, A., Limonad, L., and Montali, M., editors, *Proceedings of the 3rd International Workshop on Process Management in the AI Era (PMAI 2024) co-located with 27th European Conference on Artificial Intelligence (ECAI 2024), Santiago de Compostela, Spain, October 19, 2024*, volume 3779 of *CEUR Workshop Proceedings*, pages 11–22. CEUR-WS.org.

References VII


Vespa, M., Bellodi, E., Chesani, F., Loreti, D., Mello, P., Lamma, E., Ciampolini, A., Gavanelli, M., and Zese, R. (2024b).
Probabilistic traces in declarative process mining.
In Artale, A., Cortellessa, G., and Montali, M., editors, *AIxIA 2024 -Advances in Artificial Intelligence-XXIIIrd International Conference of the Italian Association for Artificial Intelligence, AIxIA 2024, Bolzano, Italy, November 25-28, 2024 Proceedings*, volume 15450 of *Lecture Notes in Computer Science*, pages 330–345. Springer.


Vespa, M., Bellodi, E., Loreti, D., and Chesani, F. (2026).
A preliminary approach to probabilistic process discovery.
In *Proceedings of the 41st Italian Conference on Computational Logic, June 23-25 2026, Ferrara, Italy. To appear.*, CEUR Workshop Proceedings. CEUR-WS.org.