Binary Discovery of Declarative Business Processes with ASP Preferences

F. Chesani[1], C. Di Francescomarino[2], C. Ghidini[3], D. Loreti[1], F. M. Maggi[4], P. Mello[1], M. Montali[4], S. Tessaris[4]

Context: Process Discovery in BPM

- Given a log, Process Discovery aims to learn a representation (a ***model***) of the business process underlying the log
- Different modelling languages: from procedural-like (usually closed models), to more declarative ones (usually open models, such as in **Declare**)
- Basic assumptions: executions traces in the log are good examples of the process, and are statistically relevant of the process itself
- In other words, **one-class supervised learning techniques**

Process Discovery as a two-class supervised learning

Historically, the two classes of traces are named ***positive*** and ***negative examples***.

1. In real use cases it is common to encounter positive and negative process executions
2. There is a need for balancing criteria like accuracy and recall of the learned model
3. There is a need for discovering discriminative models, i.e. what distinguish a set of executions from another (positive and negative are just labels).

Contribution

NegDis, a framework for discovery of declarative, discriminative models

- Models expressed in the Declare laaguage, with underlying formal semantics in LTL_f
- Models identified through Satisfiability-based AI techniques
- Supports user preferences (1) over templates included in the resulting model, and (2) over which process activities will be subjected to constraints
- Supports heuristics about more general/more specific models, and simplicity
- A step towards the management of the "unseen behaviors": allows to better control which behaviors should be accepted

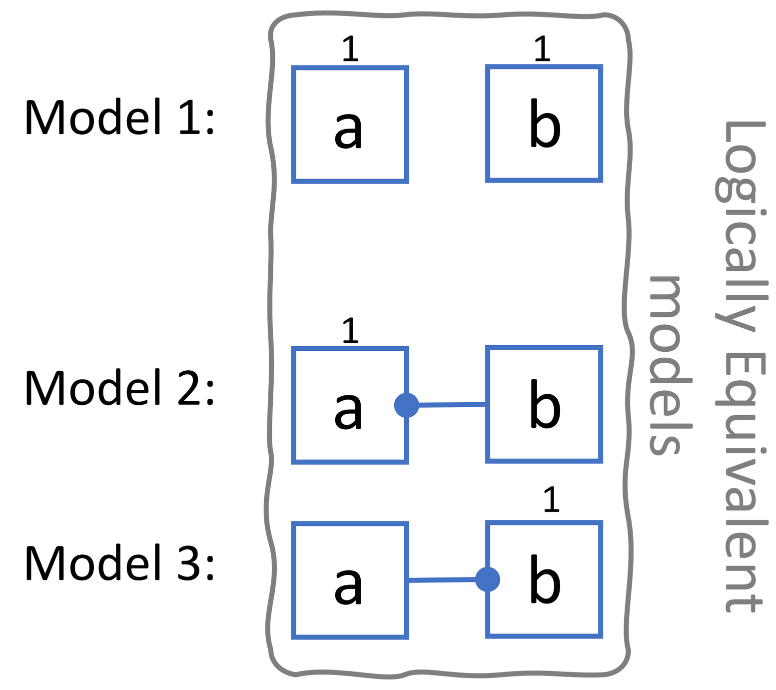
Why preferences? Why heuristics?

$L^+ = \{ <a, b>, <b, a> \}$

$L^- = \{ <a>, <b> \}$

What differentiate the traces?

- Model 1 suggests "no relation between a and b"
- Model 2 and 3 suggest there is a relation



Only the domain expert knows…

Sketch of NegDis learning algorithm

- Let **L** be a log of execution traces partitioned into two sets **L+** and **L-**
- Let be **A** the set of activities, and **D** the set of Declare templates

1. generate the set **D[A]** of templates grounded over **A**
2. Identifies the constraints **compatibles**, i.e. those constraints that accepts all the traces in **L+**
3. For each trace **t** in **L-**, identify among the compatibles the ***sheriffs***, i.e. those constraints that rejects **t**
4. Choose a model (as a conjunction of sheriffs) such that all traces in **L-** are rejected, keeping into account subsumption relation among constraints

When creating the model, several criteria can be considered:

a) generality/specifity
b) simplicity
c) user preferences over templates
d) user preferences over involved activities
e) a combination of the above

Everything implemented through ASP and Asprin: finding the model that better satisfies the criteria above becomes an optimization problem.

Perfomance evaluation of NegDis: logs size (right), and performances (below)

F. Chesani et al., "Process Discovery on Deviant Traces and Other Stranger Things," in IEEE Transactions on Knowledge and Data Engineering, doi: 10.1109/TKDE.2022.3232207.

Dataset	Log	Trace #	Activity #	Label	Positive Trace #	Negative Trace #
$BPIC12_{CANC}$				O_CANCELLED occurs	2660	10427
$BPIC12_{mean}$	BPIC12	13087	36	mean duration	8160	4927
$BPIC12_{median}$				median duration	6544	6543
$CERV_{compl}$				compliant	55	102
$CERV_{mean}$	CERV	157	16	mean duration	93	64
$CERV_{median}$				median duration	92	65
$DREYERS_{reset}$				reset executions	492	208
$DREYERS_{mean}$	DREYERS	700	33	mean duration	206	494
$DREYERS_{median}$				median duration	406	294
$PROD_{rej}$				reset executions	103	117
$PROD_{mean}$	PROD	220	26	mean duration	148	72
$PROD_{median}$				median duration	110	110
$SEPSIS_{\phi_1}$				ϕ_1	685	365
$SEPSIS_{mean}$	SEPSIS	1050	16	mean duration	838	212
$SEPSIS_{median}$				median duration	525	525
$TRAFFIC_{paid}$				fine paid	70 602	59 013
$TRAFFIC_{mean}$	TRAFFIC	129 615	10	mean duration	70 585	59 030
$TRAFFIC_{median}$				median duration	65 003	64 612

	SUBSET			CARDINALITY			Required Time (s)			
Dataset	Number of models	Min model size	Violated L^- trace %	Number of models	Min model size	Violated L^- trace %	Comp.	*sheriffs*	SUBSET	CARDINALITY
$BPIC12_{CANC}$	max	1	100%	2	1	100%	4.92	12.39	22.15	4
$BPIC12_{mean}$	max	13	0.91%	max	13	0.91%	2.83	5.87	19.99	0.08
$BPIC12_{median}$	max	23	32.71%	max	23	32.71%	2.13	7.24	6.59	54.24
$CERV_{compl}$	max	4	100%	max	4	100%	0.4	0.06	19.24	0.05
$CERV_{mean}$	max	2	100%	max	2	100%	0.01	0.91	10.86	7.79
$CERV_{median}$	max	1	100%	1	1	100%	0.01	0.84	11.8	0.5
$DREYERS_{reset}$	max	8	98.56%	4	8	98.56%	1.62	0.1	78.03	0.12
$DREYERS_{mean}$	max	4	100%	max	4	100%	0.6	2.56	23.77	462.09
$DREYERS_{median}$	max	14	96.94%	1	14	96.94%	1.25	1.07	81.16	timeout
$PROD_{rej}$	max	21	40.17%	1	21	40.17%	0.8	0.41	7.03	timeout
$PROD_{mean}$	max	21	81.94%	1	21	81.94%	0.71	0.66	8	timeout
$PROD_{median}$	max	30	78.18%	1	30	78.18%	0.56	0.93	12.31	timeout
$SEPSIS_{\phi_1}$	3	3	0.82%	2	3	0.82%	0.67	0.32	0.37	0.05
$SEPSIS_{mean}$	2	8	4.25%	2	8	4.25%	0.72	0.26	0.4	0.06
$SEPSIS_{median}$	max	14	26.86%	max	14	26.86%	0.46	0.62	2.99	0.12
$TRAFFIC_{paid}$	1	1	0.01%	1	1	0.01%	20.51	20.82	0.26	0.07
$TRAFFIC_{mean}$	max	13	0.76%	2	13	0.76%	18.98	23.4	5.63	0.05
$TRAFFIC_{median}$	max	13	0.75%	4	13	0.75%	15.35	28.03	2.02	0.04

[1]University of Bologna – [2]University of Trento – [3]FBK Trento – [4]Free University of Bozen