Process discovery using Petri net synthesis

Mathieu Poirier

supervised by:
Adrian Puerto Aubel and Éric Badouel
at:
INRIA Rennes

July 9, 2019

Outline

Outline

Process mining

Data are logs

Process mining

Data are logs

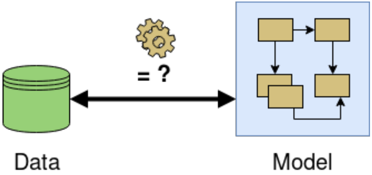

Figure: Process conformance

Process mining

Data are logs

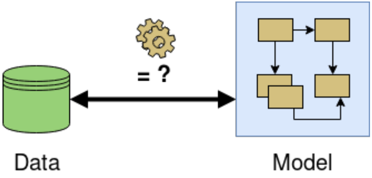

Figure: Process conformance

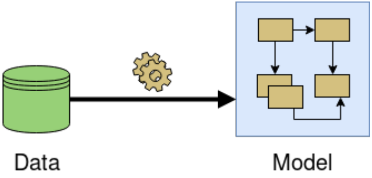

Figure: Process discovery

Outline

1. Process mining
2. Petri nets
3. From logs to a Petri net
4. New and future work

Petri nets dynamics

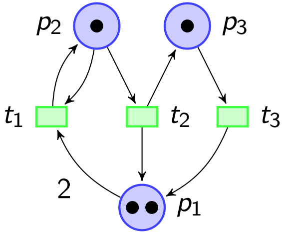

Figure: Transition activation

Petri net:

$PN = (P, T, \mathcal{F}, M_0)$

$\mathcal{F} : P \times T \cup T \times P \to \mathbb{N}$

$M_0 : P \to \mathbb{N}$

Petri nets dynamics

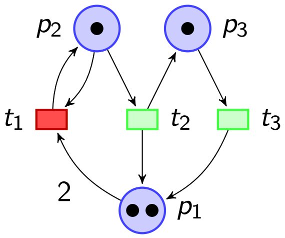

Figure: Transition activation

Petri net:

$PN = (P, T, \mathcal{F}, M_0)$

$\mathcal{F} : P \times T \cup T \times P \to \mathbb{N}$

$M_0 : P \to \mathbb{N}$

t is enabled from M:

$\forall p \in P, M(p) \geq \mathcal{F}(p, t)$

Petri nets dynamics

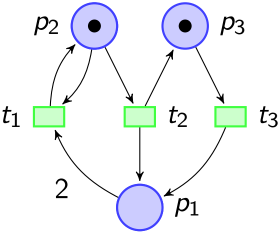

Figure: Transition activation

Petri net:
$PN = (P, T, \mathcal{F}, M_0)$
$\mathcal{F} : P \times T \cup T \times P \to \mathbb{N}$
$M_0 : P \to \mathbb{N}$
t is enabled from M:
$\forall p \in P, M(p) \geq \mathcal{F}(p, t)$

Petri nets dynamics

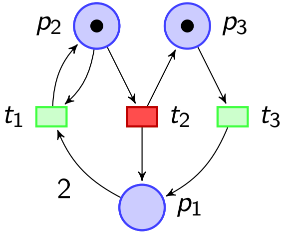

Figure: Transition activation

Petri net:

$PN = (P, T, \mathcal{F}, M_0)$

$\mathcal{F} : P \times T \cup T \times P \to \mathbb{N}$

$M_0 : P \to \mathbb{N}$

t is enabled from M:

$\forall p \in P, M(p) \geq \mathcal{F}(p, t)$

Petri nets dynamics

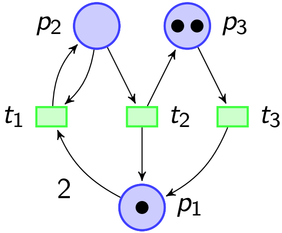

Figure: Transition activation

Petri net:
$PN = (P, T, \mathcal{F}, M_0)$
$\mathcal{F} : P \times T \cup T \times P \rightarrow \mathbb{N}$
$M_0 : P \rightarrow \mathbb{N}$
t is enabled from M:
$\forall p \in P, M(p) \geq \mathcal{F}(p, t)$

Petri Net as a process model

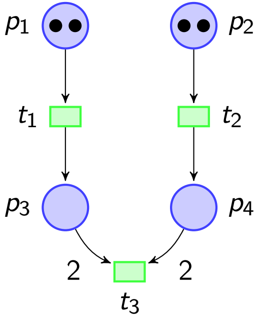

Figure: Parallelization and synchronization

Petri Net as a process model

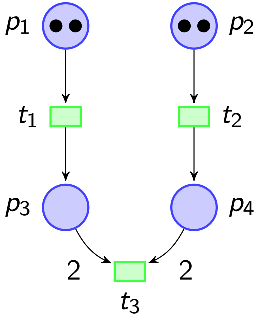

Figure: Parallelization and synchronization

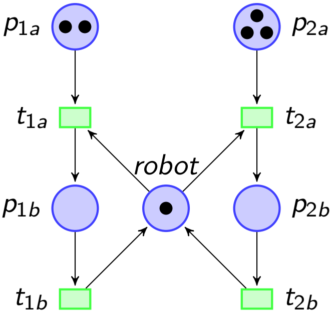

Figure: Resource sharing

Places as vectors

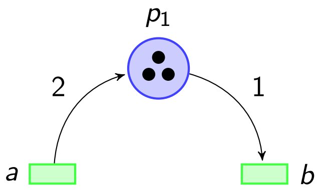

Figure: Rational places

Logs give the n transitions

Places are vectors in $\mathbb{N}^{1+2n}$:
$p = (p_0; pre(T); post(T))$

Places as vectors

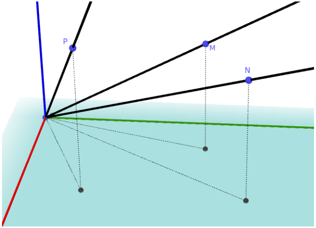

Figure: Set of places in their space

Logs give the n transitions

Places are vectors in $\mathbb{N}^{1+2n}$:
$p = (p_0; pre(T); post(T))$

Places as vectors

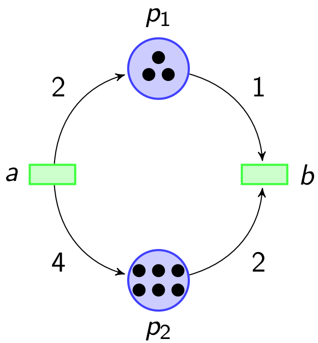

Figure: Rational places

Logs give the n transitions

Places are vectors in $\mathbb{N}^{1+2n}$:
$p = (p_0; pre(T); post(T))$

Places as vectors

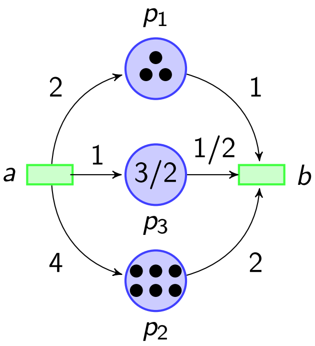

Figure: Rational places

Logs give the n transitions

Places are vectors in $\mathbb{N}^{1+2n}$:
$p = (p_0; pre(T); post(T))$

Places are constraints

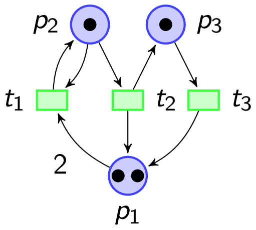

Figure: Petri net with 3 transitions

Each place p is a constraint

Places are constraints

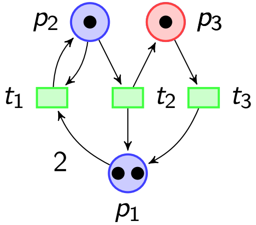

Figure: Petri net with 3 transitions

Constraint from p_3:
$n_3 \leq 1 + n_2$

Each place p is a constraint

Places are constraints

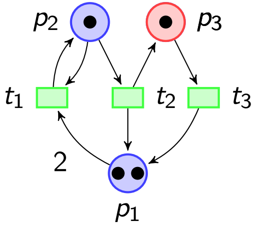

Figure: Petri net with 3 transitions

Constraint from p_3:
$n_3 \leq 1 + n_2$

Each place p is a constraint:

$$\mathscr{L}(PN) = \bigcap_{p \in P} \mathscr{L}(PN(p))$$

Outline

Specification

- Replay-fitness: Logs included

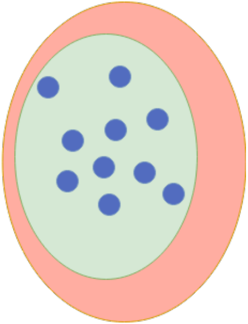

Figure: Replay-fitness, generalization and precision

Specification

- Replay-fitness: Logs included
- Generalization: more than the Logs

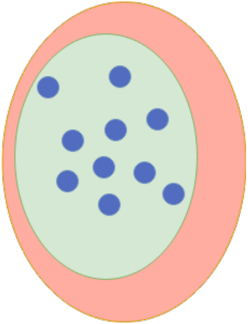

Figure: Replay-fitness, generalization and precision

Specification

- Replay-fitness: Logs included
- Generalization: more than the Logs
- Precision: not too much

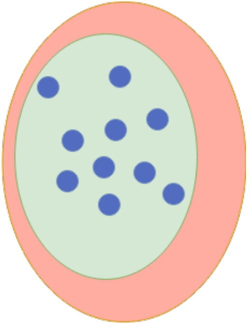

Figure: Replay-fitness, generalization and precision

Specification

- Replay-fitness: Logs included
- Generalization: more than the Logs
- Precision: not too much
- Size: as low as possible

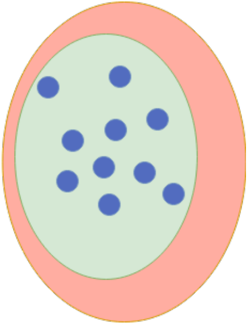

Figure: Replay-fitness, generalization and precision

Specification

- Replay-fitness: Logs included
- Generalization: more than the Logs
- Precision: not too much
- Size: as low as possible

Generalization: closure as Petri Net language of the set of Logs

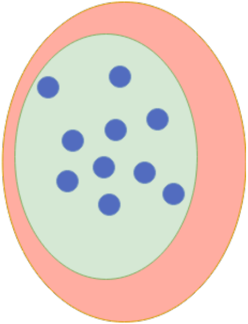

Figure: Replay-fitness, generalization and precision

From replay-fitness to cone of regions

Places are restrictions. Need to restrict them for replay-fitness
$w \cdot e \in \mathcal{L}(PN) \iff \forall p \in P, \langle p, \vec{v}(w \cdot e) \rangle \geq 0$

From replay-fitness to cone of regions

Places are restrictions. Need to restrict them for replay-fitness

$w \cdot e \in \mathcal{L}(PN) \iff \forall p \in P, \langle p, \vec{v}(w \cdot e) \rangle \geq 0$

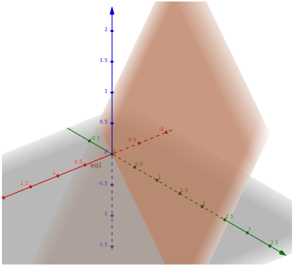

Figure: Half-space

From replay-fitness to cone of regions

Places are restrictions. Need to restrict them for replay-fitness

$w \cdot e \in \mathcal{L}(PN) \iff \forall p \in P, \langle p, \vec{v}(w \cdot e) \rangle \geq 0$

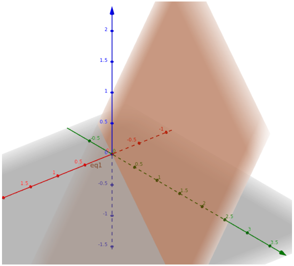

Figure: Half-space

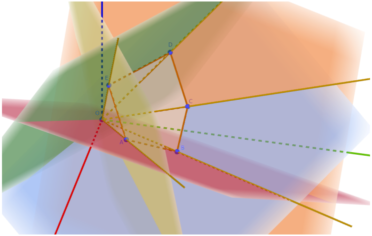

Figure: Cone of regions

From replay-fitness to cone of regions

Places are restrictions. Need to restrict them for replay-fitness

$w \cdot e \in \mathcal{L}(PN) \iff \forall p \in P, \langle p, \vec{v}(w \cdot e) \rangle \geq 0$

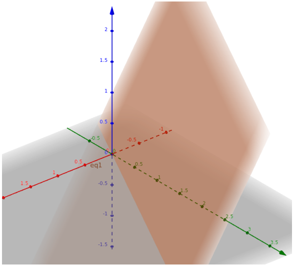

Figure: Half-space

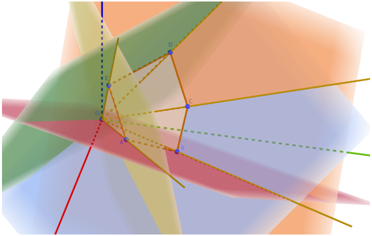

Figure: Cone of regions

Extremal rays have a big role

Precision thanks to places

Places are restrictions. They help for precision !

$w \cdot e \notin \mathcal{L}(PN) \iff \exists p \in P, \langle p, \vec{v}(w \cdot e) \rangle < 0$

Precision thanks to places

Places are restrictions. They help for precision !

$w \cdot e \notin \mathcal{L}(PN) \iff \exists p \in P, \langle p, \vec{v}(w \cdot e) \rangle < 0$

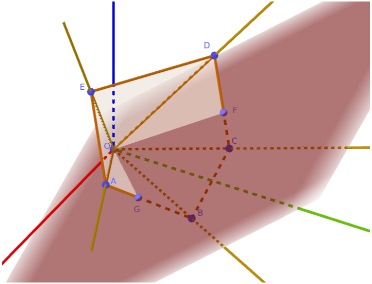

Figure: Separation problem

Precision thanks to places

Places are restrictions. They help for precision !
$w \cdot e \notin \mathcal{L}(PN) \iff \exists p \in P, \langle p, \vec{v}(w \cdot e) \rangle < 0$

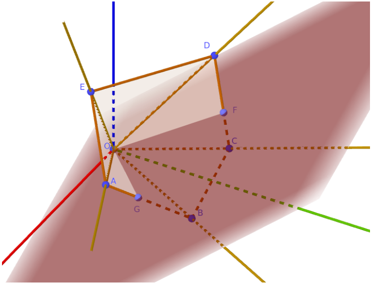

Figure: Separation problem

Extremal rays are a complete set of regions

Synthesized Petri Net extracted from them

Outline

1. Process mining
2. Petri nets
3. From logs to a Petri net
4. New and future work

New approach about size

Extremal rays haven't the lowest size

New approach about size

Extremal rays haven't the lowest size

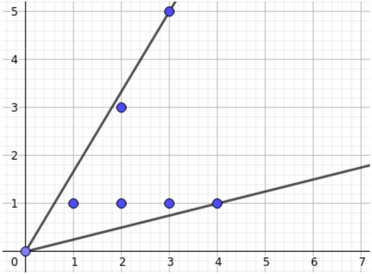

Figure: Hilbert basis of a 2D cone

Hilbert basis should be better

Computed through Elliot-McMahon algorithm

Outlier detection

Presence of outliers redefine replay-fitness

Outlier detection

Presence of outliers redefine replay-fitness

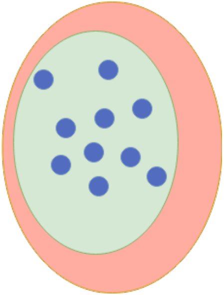

Figure: Replay-fitness as strict criterion

Outlier detection

Presence of outliers redefine replay-fitness

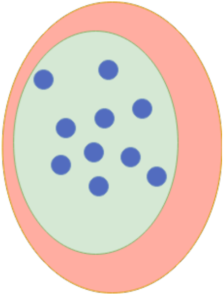

Figure: Replay-fitness as strict criterion

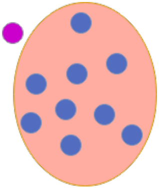

Figure: Soft replay-fitness allows to identify outliers

Conclusion

On-going work

- Check Hilbert Bases' properties
- Address outlier detection

Conclusion

On-going work

- Check Hilbert Bases' properties
- Address outlier detection

Thanks!