Traceability and Concurrent Fairness in Petri Nets

Joanna Jółkowska & Edward Ochmański

Nicolaus Copernicus University, Toruń, Poland

DNTTT'08 • Cremona • 9-11 October 2008

Transition system

Transition system – triple $S=(A,Q,q_0)$

- A – finite set of **actions**
- Q – countable set of **states**
  - state q – partial function $q:A\to Q$
  - extension $q:A^*\to Q$ $q(uv)=q(u)(v)$
- q_0 – initial state

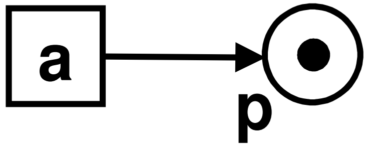
Sequential behaviour of transition systems

Given a transition system $S=(A,Q,q_0)$

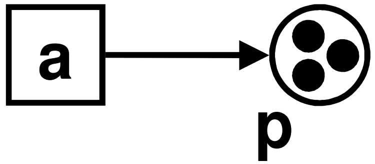
- **computation** of S:
  – any sequence $a_1a_2a_3\ldots$ s.t. $(\forall i>0)\ q_{i-1}(a_i)=q_i$
- **behaviour** (or language) of S:
  – set of all computations $L^\infty(S)=L(S)\cup L^\omega(S)$,
  where $L(S)$ – finite computations
  $L^\omega(S)$ – infinite computations

Petri nets

- Elementary nets (capacity=1)

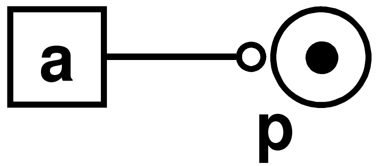


- Place/transistion nets (capacity=∞)

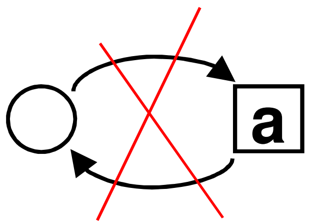


- Inhibitor nets (a enabled if p empty)

- Reset nets (a cleans the place p)



- Pure net = net without self-loops



Part I

INDEPENDENT ACTIONS

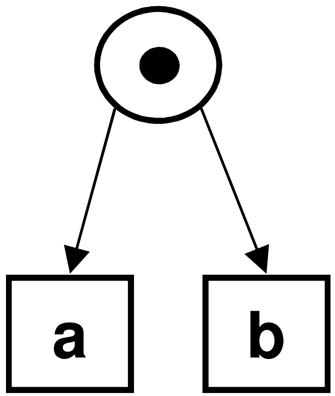
Independency induced by a system

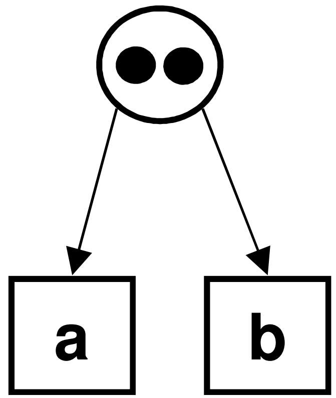
$S=(A,Q,q_0)$ – transition system; state $q \in Q$;

actions $a,b \in A$ are:

locally independent in q	$a I_q b$	$q(ab)=q(ba)$
locally dependent in q	$a D_q b$	$q(ab) \neq q(ba)$
locally concurrent in q	$a \|\|_q b$	$(\exists q' \in Q)\ q(ab)=q(ba)=q'$
independent in S	$a I_S b$	$(\forall q \in Q)\ a I_q b$
dependent in S	$a D_S b$	$(\exists q \in Q)\ a D_q b$

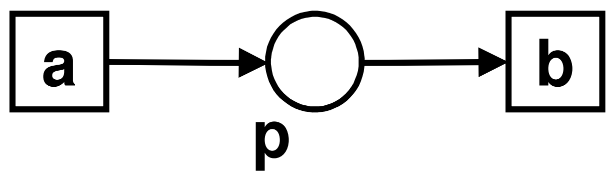
Local independency – examples

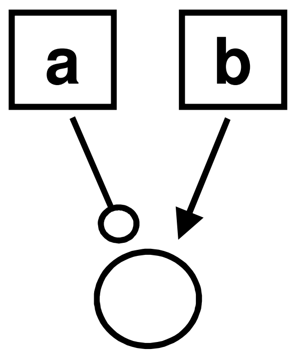




- Locally independent but not concurrent
- Locally independent and concurrent

Local dependency – examples





- Dependent in M(p)=0:
  - Mab, but not Mba
- Concurrent in M(p)>0:
  - Mab and Mba

- Dependent in M(p)=0:
  Mab i Mba, but Mab≠Mba

Independency induced by a language

$L \subseteq A^*$ - prefix-closed language

$a,b \in A^*$ - letters (actions)

I_L – independency induced by L:

$$aI_Lb \text{ iff } (\forall u,v \in A^*)\ uabv \in L \Leftrightarrow ubav \in L$$

ab and ba are syntactically equivalent

Independency induced by a language – examples

- L_1={abc,bac}   aIb
- L_2={ab}   aDb
  (since ab$\in$L and ba$\notin$L)
- L_3={abc,bad}   aDb
  (since abc$\in$L and bac$\notin$L)

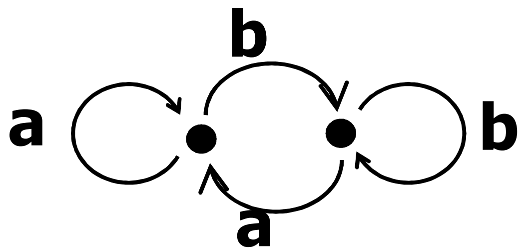
Comparison of two kinds of independencies

- For any transition system $S=(A,Q,q_0)$:

$$I_S \subseteq I_{L(S)}$$

- and the inclusion may be strict:

S:



$I_S=\varnothing$

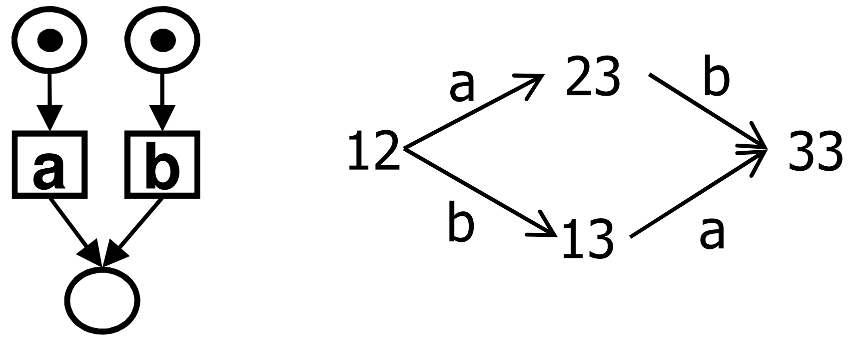
$L(S)=(a\cup b)^*$

$I_L=\{(a,b)\}$

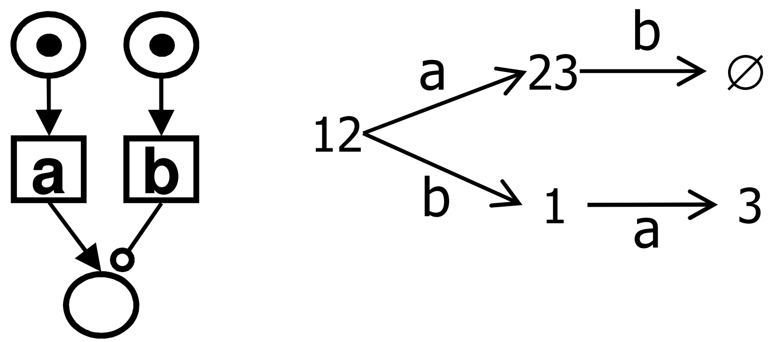
Diamond property

$$(\forall q)\ (\forall a,b)\ q(ab)=q' \wedge q(ba)=q'' \Rightarrow q'=q''$$

- Elementary, p/t, inhibitor nets have diamond property



- Reset nets have not diamond property



When $I_S = I_{L(S)}$?

- **Theorem:** If a transition system S is diamond, then independencies induced by the system S and the language L(S) coincide:

$$I_S = I_{L(S)}$$

- **Corollary:** The equality holds in elementary nets, p/t nets and inhibitor nets.
- **Remark:** It does not hold in reset nets.

Computing independency

The problem „Are two actions dependent?" is

- **decidable**
  for elementary and place/transition nets.
- **undecidable**
  for inhibitor and reset nets.

Some known decision problems

Coverability problem Net N, state M → „Is there a reachable state M′ such that M′≥M?”
Reachability problem Net N, state M → „Is M reachable in N?”
Place emptiness problem Net N, place p → „Is there a reachable state M such that M(p)=0?”

All three problems: Decidable for p/t nets

Reachability problem and place emptiness problem: Undecidable for reset nets and inhibitor nets

Decidability of dependency in place/transition nets

Problem

Net N, state M, place p → "Is there a reachable state M' such that M'≥M and M'(p)=0?"

is **decidable** for p/t nets.

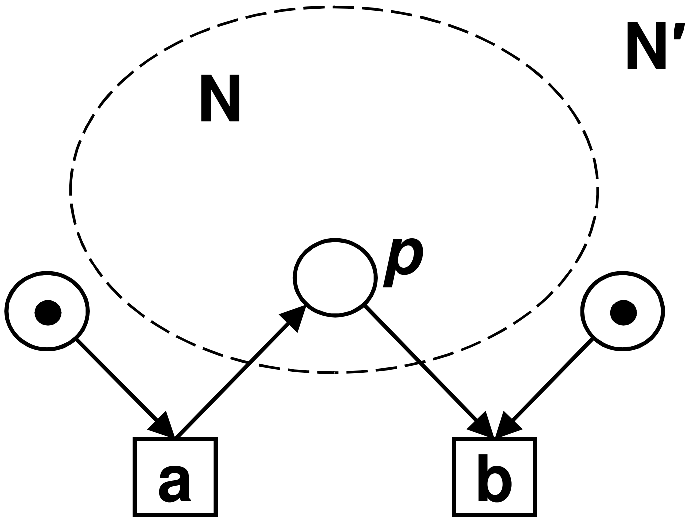
Problem

Net N, actions a,b → "Are a,b dependent?"

is **decidable** for p/t nets.

Undecidability of dependency in inhibitor and reset nets

Place emptiness undecidable	⇒	Dependency undecidable



A state with empty *p*
is reachable in N
iff
actions a, b
are dependent in N'

Part II

TRACEABILITY

From sequential to concurrent behaviour

Transition system $S=(A,Q,q_0)$, behaviour $L^\infty(S)$

- independency: aI_Sb iff $(\forall q,q')\ q(ab)=q' \Leftrightarrow q(ba)=q'$
- dependency: aDb iff non aI_Sb

Equivalence of computations $u,v \in L^\infty(S)$:

$$u \approx v \quad \text{iff} \quad (\forall a,b \in A)\ aDb \Rightarrow \pi_{a,b}(u) = \pi_{a,b}(v)$$

Trace (process):

set of equivalent computations $[u]=\{v \in L^\infty(S);\ u \approx v\}$

Concurrent behaviour of S :

set of traces (processes) $[L^\infty(S)]=T^\infty(S)$

Traceability

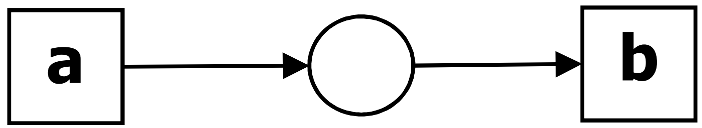
Transition system S is **traceable** iff

$$(\forall a,b\in A)\ ((\exists q\in Q)\ a||_q b\ ) \Rightarrow aI_S b$$

Transition system S is **not traceable** iff

$$(\exists a,b\in A)\ ((\exists q\in Q)\ a||_q b) \wedge ((\exists q'\in Q)\ aD_{q'} b)$$

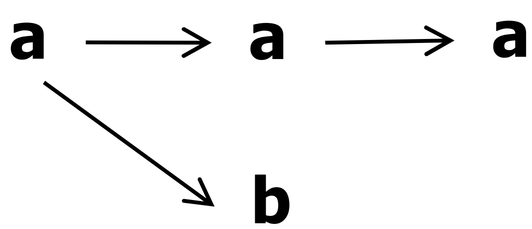
Traceability – motivations



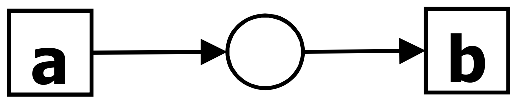
aDb because aD_Mb for a state M=[0]

Trace [aaab]: **a** → **a** → **a** → **b**

True behaviour [aaab]:

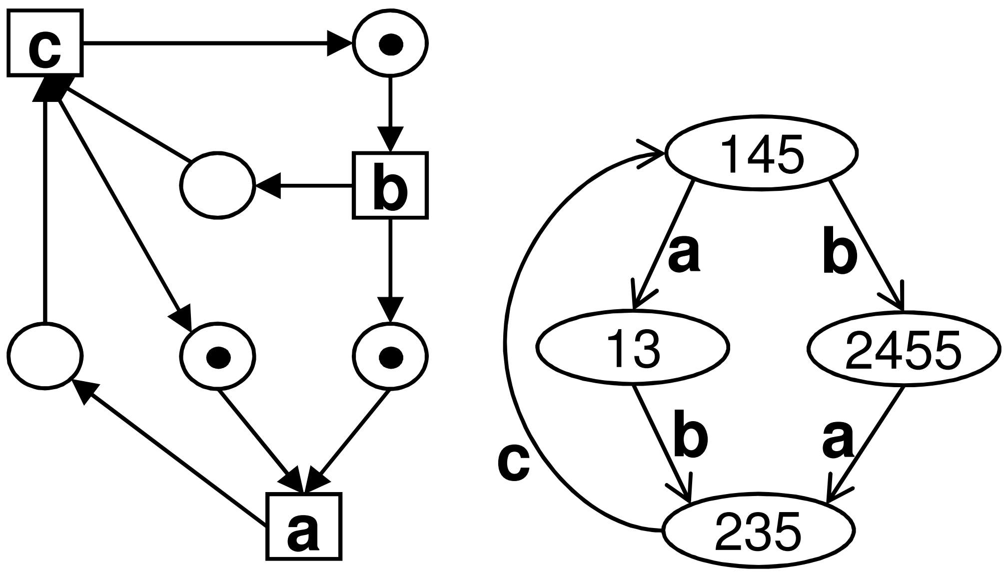


Traceability – examples



Not traceable net:

$aD_{[0]}b$ and $a||_{[1]}b$



Traceable net: aIb

Decidability of traceability for place/transition nets

Problem

Net N, actions a,b $\rightarrow$ "Is there a reachable state M such that $a||_M b$?"

is **decidable** for place/transition nets.

+ decidability of dependency

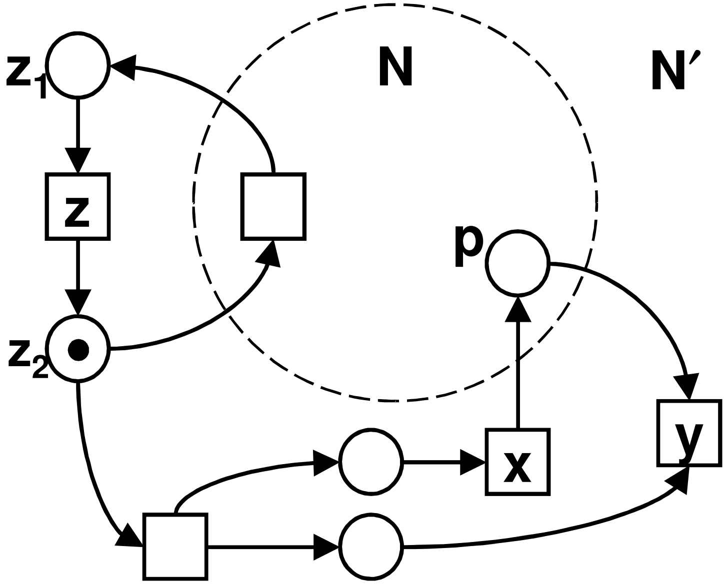
Problem

Net N $\rightarrow$ "Is the net traceable?"

is **decidable** for place/transition nets.

Undecidability of traceability for inhibitor and reset nets

Place emptiness is undecidable	⇒	Traceability is undecidable



If $M_0(p)\neq 0$, then

A state M with M(p)=0 is reachable in N
iff
actions x,y are locally concurrent (and dependent) in N′
iff
N′ is not traceable

Part III

CONCURRENT FAIRNESS

Ethics of sequential computations

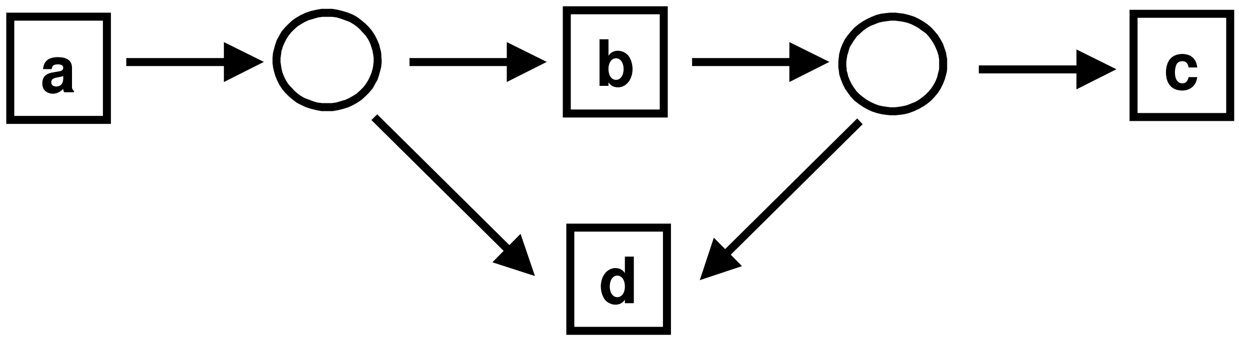
An infinite computation $w \in L^{\omega}(S)$ is:

- **just** iff any action **permanently enabled** in w
  occurs infinitely often in w
- **fair** iff any action **infinitely often enabled** in w
  occurs infinitely often in w
- **superfair** iff any action **live** in w
  occurs infinitely often in w

For finite computations $w \in L(S)$ all three notions coincide:

- w is just iff w is fair iff w is super fair
  iff w is non-extendable

Sequential fairness hierarchy



SFair	$\subset$	Fair	$\subset$	Just
⟒		⟒		⟒
$(abadabc)^\omega$		$(abc)^\omega$		$ab\,(acb)^\omega$

Ethics of concurrent processes

An infinite process $\tau \in T^{\omega}(S)$ is:

- **universally just** iff any computation $w \in \tau$ is just
- **existentially just** iff there is a just computation $w \in \tau$
- **universally fair** iff any computation $w \in \tau$ is fair
- **existentially fair** iff there is a fair computation $w \in \tau$
- **universally superfair** iff any computation $w \in \tau$ is superfair
- **existentially superfair** iff there is a superfair computation $w \in \tau$

Process fairness hierarchy

eSFAIR	$\subseteq$	eFAIR	$\subseteq$	eJUST
∪|		∪|		∪|
uSFAIR	$\subseteq$	uFAIR	$\subseteq$	uJUST

Always eSFAIR = uSFAIR

In arbitrary transition system

eSFAIR = uSFAIR

eSFAIR	$\subseteq$	eFAIR	$\subseteq$	eJUST
=		$\cup\!\!\mid$		$\cup\!\!\mid$
uSFAIR	$\subseteq$	uFAIR	$\subseteq$	uJUST

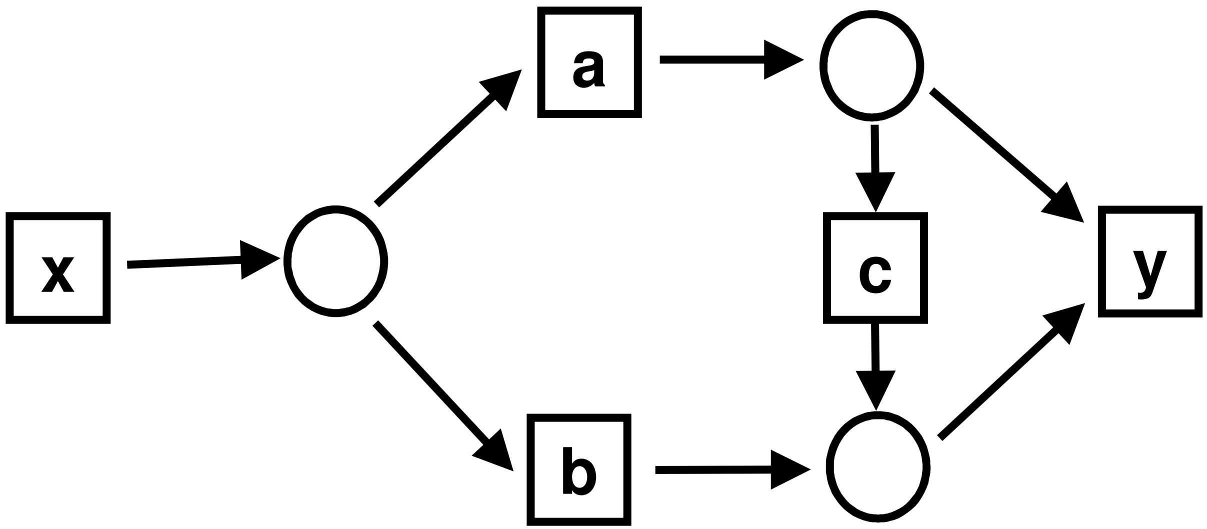
Process fairness hierarchy

$$
\begin{array}{ccccc}
 & & \text{eFAIR} & \subseteq & \text{eJUST} \\
 & & \rotatebox{90}{$\subseteq$} & & \rotatebox{90}{$\subseteq$} \\
\text{SFAIR} & \subseteq & \text{uFAIR} & \subseteq & \text{uJUST}
\end{array}
$$

Question: Are the inclusions proper?

Strict inclusion

SFAIR $\subset$ uFAIR



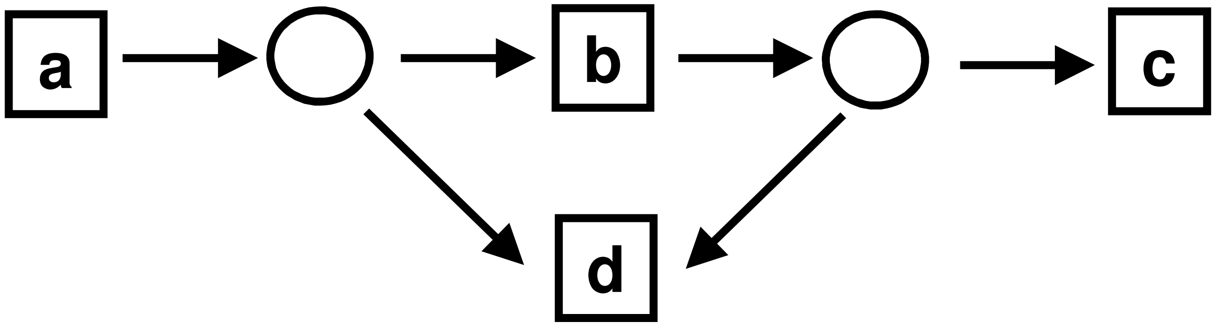
The process $[(xbxay)^{\omega}]$ is universally fair,
but not superfair

Process fairness hierarchy

		eFAIR	$\subseteq$	eJUST
		$\rotatebox{90}{$\subseteq$}$		$\rotatebox{90}{$\subseteq$}$
SFAIR	$\subset$	uFAIR	$\subseteq$	uJUST

Strict inclusion

uFAIR ⊂ eFAIR

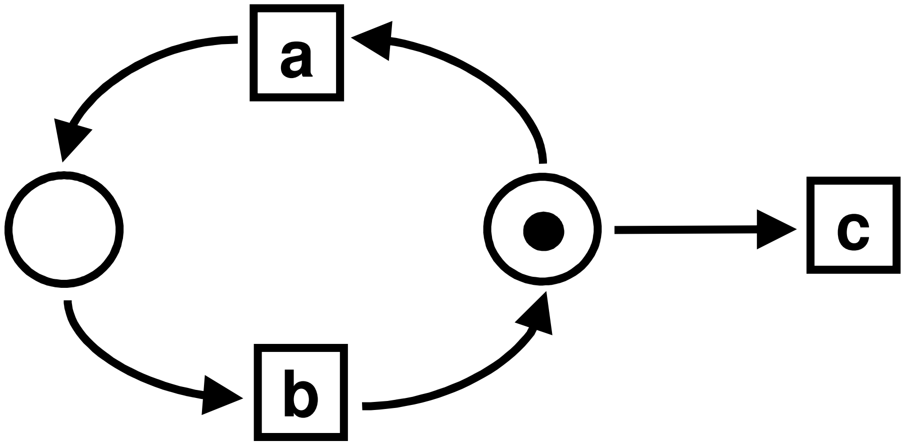


The process $[(abc)^{\omega}]=[ab(acb)^{\omega}]$ is existentially fair, but not universally fair

Process fairness hierarchy

$$\begin{array}{ccccccc} & & \text{eFAIR} & \subseteq & \text{eJUST} \\ & & \cup & & \cup\!| \\ \text{SFAIR} & \subset & \text{uFAIR} & \subseteq & \text{uJUST} \end{array}$$

Strict inclusions: eFAIR ⊂ eJUST and uFAIR ⊂ uJUST



The process $[(ab)^\omega]$ is eJUST, but not eFAIR
and uJUST, but not uFAIR

Process fairness hierarchy

$$\begin{array}{ccccc} & & \text{eFAIR} & \subset & \text{eJUST} \\ & & \cup & & \cup| \\ \text{SFAIR} & \subset & \text{uFAIR} & \subset & \text{uJUST} \end{array}$$

Last question:

Is eJUST equal to uJUST or not?

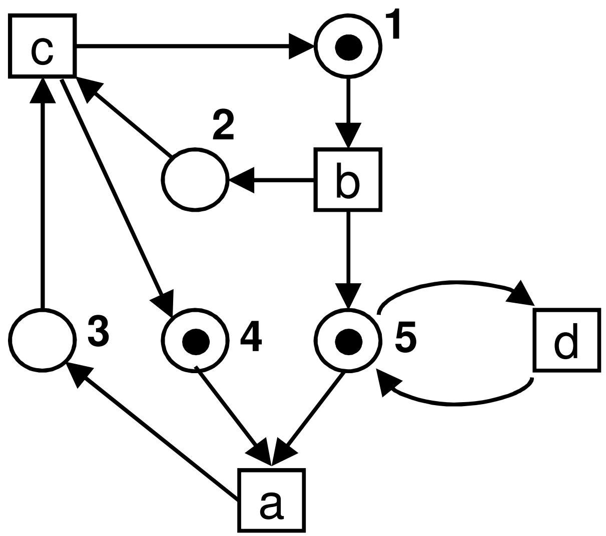
Elementary and pure p/t-nets: eJUST = uJUST

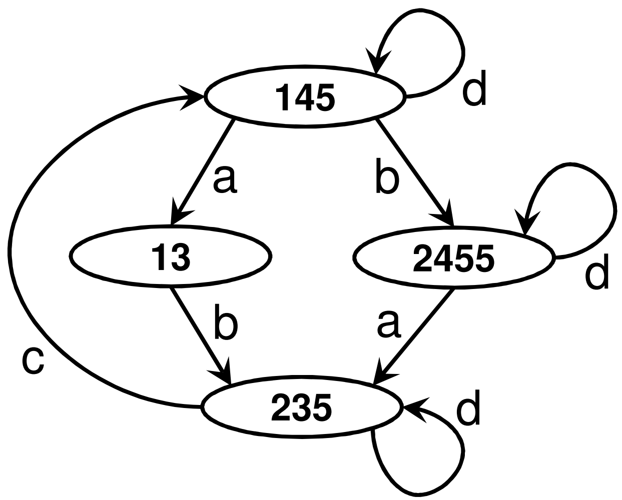
In **elementary** and **pure p/t-nets**
any existentially JUST process is universally JUST

$$
\begin{array}{ccccc}
 & & \text{eFAIR} & \subset & \text{eJUST} \\
 & & \cup & & \shortparallel \\
\text{SFAIR} & \subset & \text{uFAIR} & \subset & \text{uJUST}
\end{array}
$$

P/t-nets with self-loops

The inclusion uJUST ⊂ eJUST is proper





The process [(abc)$^{\omega}$]=[(bac)$^{\omega}$] is eJUST but not uJUST, because (abc)$^{\omega}$ is just, and (bac)$^{\omega}$ is not just

The case graph shows that aIb, thus the computations (abc)$^{\omega}$ and (bac)$^{\omega}$ are equivalent

Strict hierarchy in general p/t-nets

$$
\begin{array}{ccccc}
 & & \text{eFAIR} & \subset & \text{eJUST} \\
 & & \cup & & \cup \\
\text{SFAIR} & \subset & \text{uFAIR} & \subset & \text{uJUST}
\end{array}
$$

eFAIR and uJUST are incomparable

Non-interleaving ethics

An infinite process $\tau \in T^{\omega}(S)$ is:

- η**-just** iff any action $a \in A$ either occurs infinitely often in τ or is not permanently enabled in τ after any finite prefix of τ
- η**-fair** iff any action $a \in A$ either occurs infinitely often in τ or is nowhere enabled in τ after some finite prefix of τ
- η**-super fair** iff any action $a \in A$ either occurs infinitely often in τ or is dead in S after some finite prefix of τ

η-classes in our hierarchy

In arbitrary concurrent system:

$$
\begin{array}{ccccc}
 & & & & \eta\text{JUST} \\
 & & & & \Vert \\
 & & \text{eFAIR} & \subseteq & \text{eJUST} \\
 & & \cup\!\vert & & \cup\!\vert \\
\text{SFAIR} & \subseteq & \text{uFAIR} & \subseteq & \text{uJUST} \\
\Vert & & \Vert & & \\
\eta\text{SFAIR} & & \eta\text{FAIR} & &
\end{array}
$$

Publications

- E. Ochmański, J. Pieckowska: On Ethics of Mazurkiewicz Traces. Fundamenta Informaticae 80(1-3), pp. 259-272. IOS Press 2007.
- J. Jółkowska, E. Ochmański: On Trace-Expressible Behaviour of Petri Nets. Fundamenta Informaticae 85(1-4), pp. 281-295. IOS Press 2008.
- J. Jółkowska: Ethics of Petri Net Processes in the Light of Trace Theory (in Polish). PhD Thesis, Toruń/Warszawa, 2008.

the end