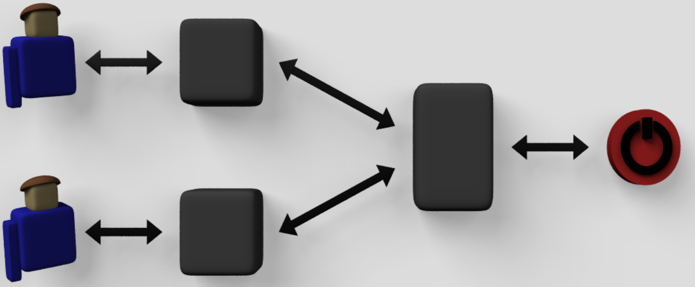
On Classes of Distributed Petri Nets

Jens-Wolfhard Schicke-Uffmann

2018-04-11

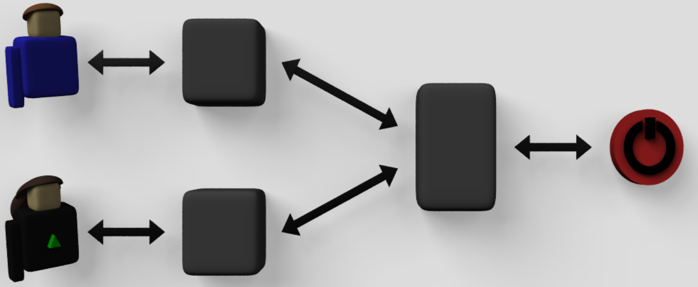
Motivating Example

Consider a network service with (wlog. 2) users:



Motivating Example

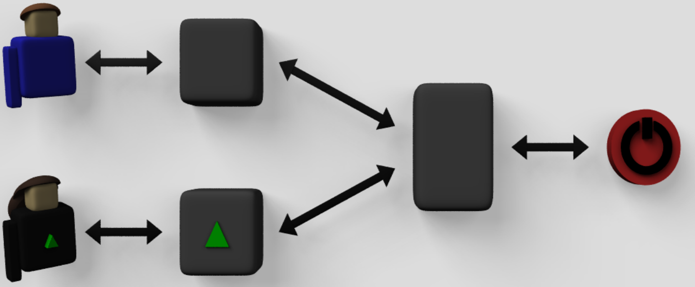
Consider a network service with (wlog. 2) users:



However, some users are hackers.

Motivating Example

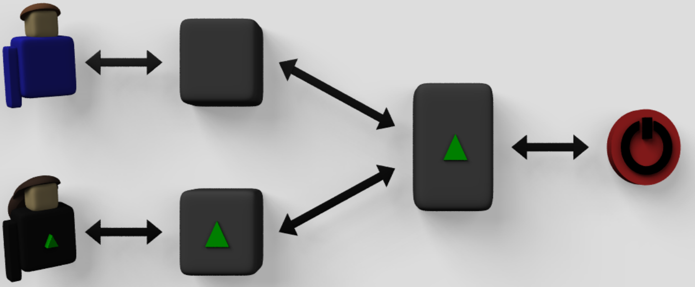
Consider a network service with (wlog. 2) users:



However, some users are hackers.

Motivating Example

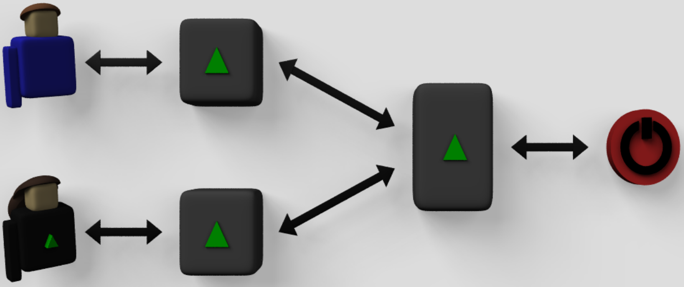
Consider a network service with (wlog. 2) users:



However, some users are hackers.

Motivating Example

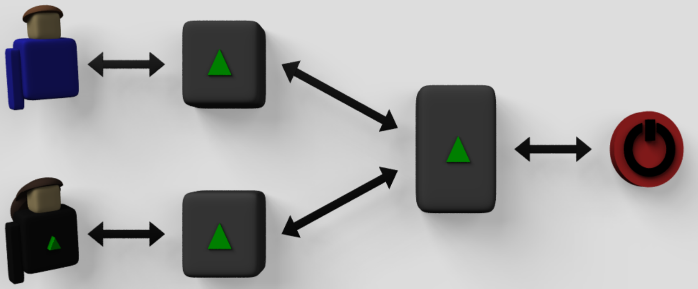
Consider a network service with (wlog. 2) users:



However, some users are hackers.

Motivating Example

Consider a network service with (wlog. 2) users:

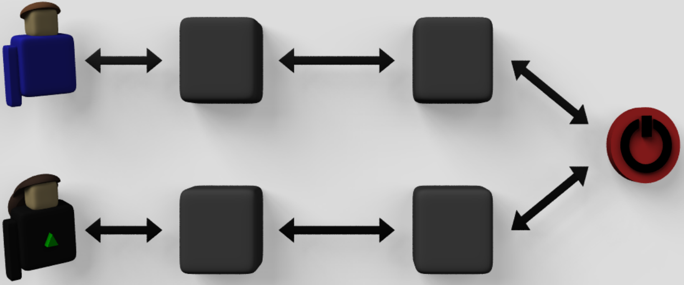


However, some users are hackers.

Abstract view: Hacker uses information flow + magic.

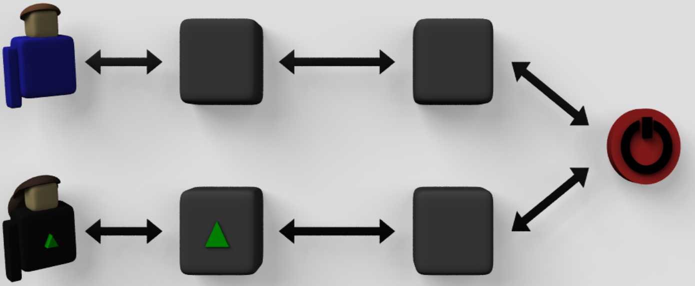
Motivating Example

Separate information flows!



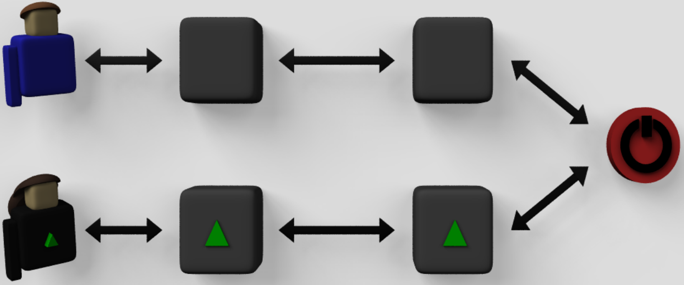
Motivating Example

Separate information flows!



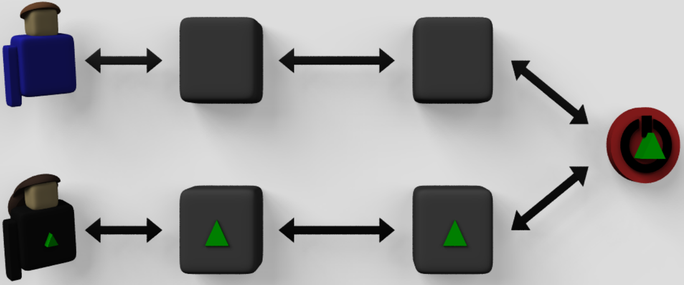
Motivating Example

Separate information flows!



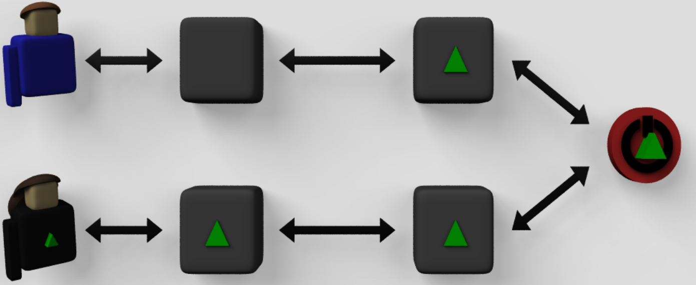
Motivating Example

Separate information flows!



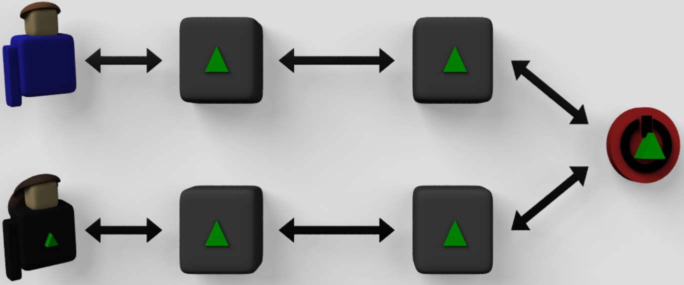
Motivating Example

Separate information flows!



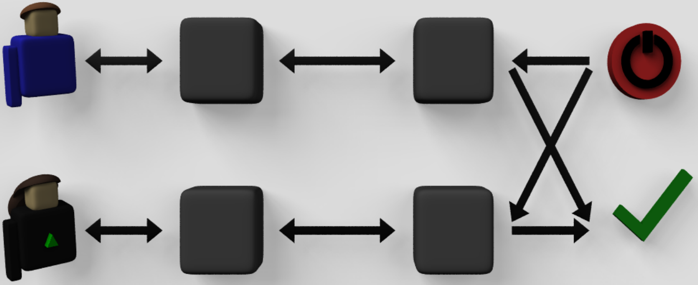
Motivating Example

Separate information flows!



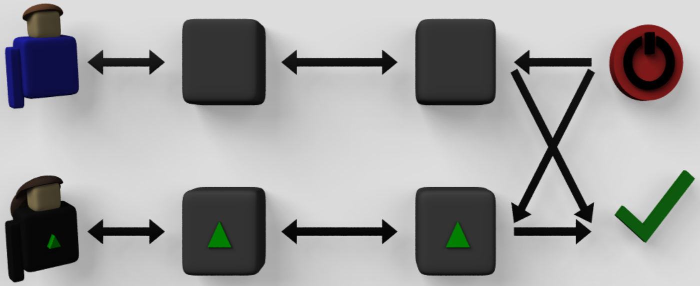
Motivating Example

Separate shutdown success reporting!



Motivating Example

Separate shutdown success reporting!



Are We Just Too Stupid?

- Maybe there is a better way?

Are We Just Too Stupid?

- Maybe there is a better way?
- Big Data?

Are We Just Too Stupid?

- Maybe there is a better way?
- Paxos?

Are We Just Too Stupid?

- Maybe there is a better way?
- Blockchain?

Are We Just Too Stupid?

- Maybe there is a better way?
- Edge Computing?

Are We Just Too Stupid?

- Maybe there is a better way?
- Use infinitely many computers?
- What exactly is an acceptable solution here?

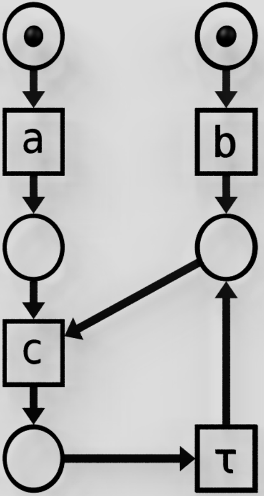
Formal Model

Abstract as much as possible. We need:

- (Parts of) the system can be in different states
- (Parts of) the system can do various things
- The parts of the system are spatially distributed
- Parts of the system send each other information

Petri nets

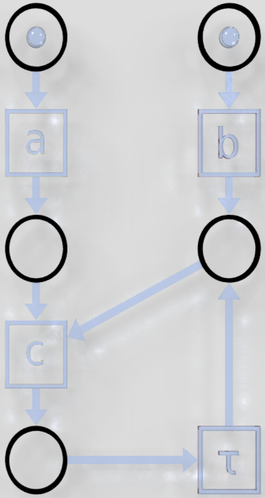
A *Petri net* is a tuple
$N = (S, T, F, M_0, \ell)$ with



Petri nets

A *Petri net* is a tuple $N = (S, T, F, M_0, \ell)$ with

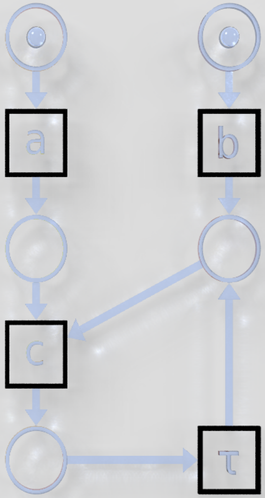
- S a set of *places*,



Petri nets

A *Petri net* is a tuple $N = (S, T, F, M_0, \ell)$ with

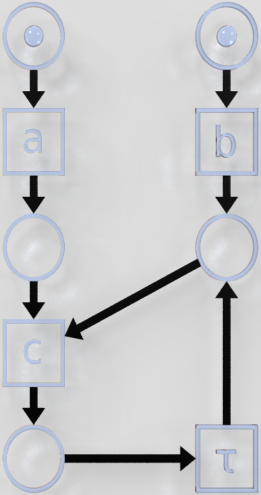
- S a set of *places*,
- T a set of *transitions* ($S \cap T = \emptyset$),



Petri nets

A *Petri net* is a tuple $N = (S, T, F, M_0, \ell)$ with

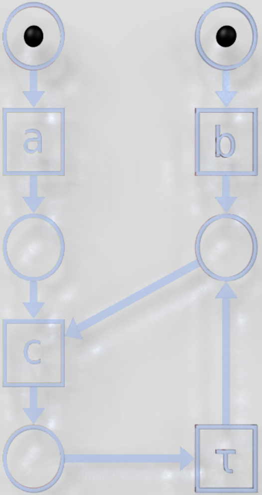
- S a set of *places*,
- T a set of *transitions* $(S \cap T = \emptyset)$,
- $F : (S \times T \cup T \times S) \to \mathbb{N}$ a *flow relation* including *arc weights*,



Petri nets

A *Petri net* is a tuple $N = (S, T, F, M_0, \ell)$ with

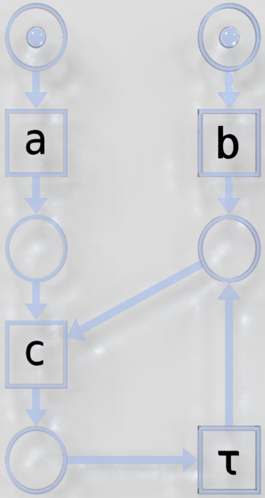
- S a set of *places*,
- T a set of *transitions* $(S \cap T = \emptyset)$,
- $F : (S \times T \cup T \times S) \to \mathbb{N}$ a *flow relation* including *arc weights*,
- $M_0 : S \to \mathbb{N}$ an *initial marking*, and



Petri nets

A *Petri net* is a tuple $N = (S, T, F, M_0, \ell)$ with

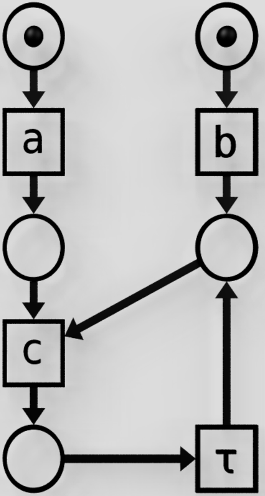
- S a set of *places*,
- T a set of *transitions* $(S \cap T = \emptyset)$,
- $F : (S \times T \cup T \times S) \to \mathbb{N}$ a *flow relation* including *arc weights*,
- $M_0 : S \to \mathbb{N}$ an *initial marking*, and
- $\ell : T \to \mathrm{Act} \cup \{\tau\}$ a *labelling function*.



Dynamic Behaviour – Firing Rule

Let $N = (S, T, F, M_0, \ell)$ be a net.

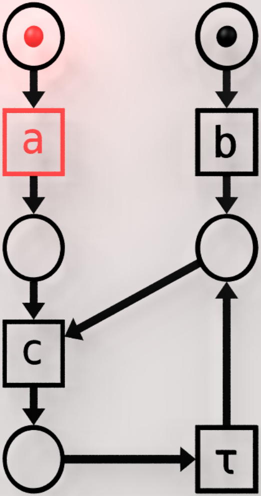
- A multiset $M \in \mathbb{N}^S$ is a *marking* of N.
- $t \in T$ is *enabled* if ${}^\bullet t \leq M$.
- A nonempty, finite $G \in \mathbb{N}^T$ is a *step* from M to marking M' iff ${}^\bullet G \leq M$ and $M' = M - {}^\bullet G + G^\bullet$.
- $[M_0\rangle$ denotes the set of *reachable* markings.
- If $[M_0\rangle \subseteq \{0, 1\}^S$ the net is *1-safe*.



Dynamic Behaviour – Firing Rule

Let $N = (S, T, F, M_0, \ell)$ be a net.

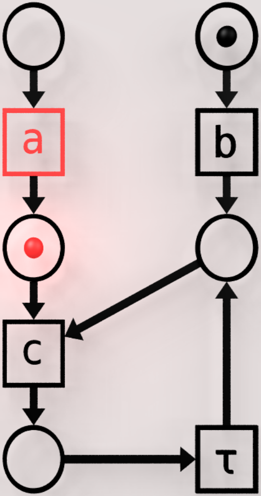
- A multiset $M \in \mathbb{N}^S$ is a *marking* of N.
- $t \in T$ is *enabled* if ${}^\bullet t \leq M$.
- A nonempty, finite $G \in \mathbb{N}^T$ is a *step* from M to marking M' iff ${}^\bullet G \leq M$ and $M' = M - {}^\bullet G + G^\bullet$.
- $[M_0\rangle$ denotes the set of *reachable* markings.
- If $[M_0\rangle \subseteq \{0, 1\}^S$ the net is *1-safe*.



Dynamic Behaviour – Firing Rule

Let $N = (S, T, F, M_0, \ell)$ be a net.

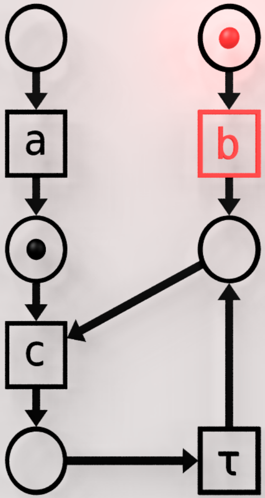
- A multiset $M \in \mathbb{N}^S$ is a *marking* of N.
- $t \in T$ is *enabled* if ${}^\bullet t \leq M$.
- A nonempty, finite $G \in \mathbb{N}^T$ is a *step* from M to marking M' iff ${}^\bullet G \leq M$ and $M' = M - {}^\bullet G + G^\bullet$.
- $[M_0\rangle$ denotes the set of *reachable* markings.
- If $[M_0\rangle \subseteq \{0, 1\}^S$ the net is *1-safe*.



Dynamic Behaviour – Firing Rule

Let $N = (S, T, F, M_0, \ell)$ be a net.

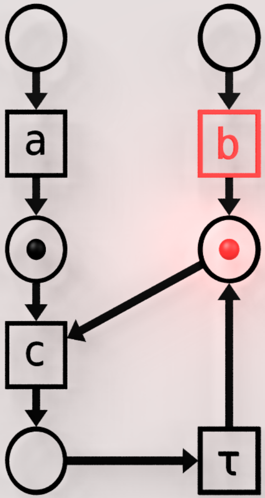
- A multiset $M \in \mathbb{N}^S$ is a *marking* of N.
- $t \in T$ is *enabled* if ${}^\bullet t \leq M$.
- A nonempty, finite $G \in \mathbb{N}^T$ is a *step* from M to marking M' iff ${}^\bullet G \leq M$ and $M' = M - {}^\bullet G + G^\bullet$.
- $[M_0\rangle$ denotes the set of *reachable* markings.
- If $[M_0\rangle \subseteq \{0, 1\}^S$ the net is *1-safe*.



Dynamic Behaviour – Firing Rule

Let $N = (S, T, F, M_0, \ell)$ be a net.

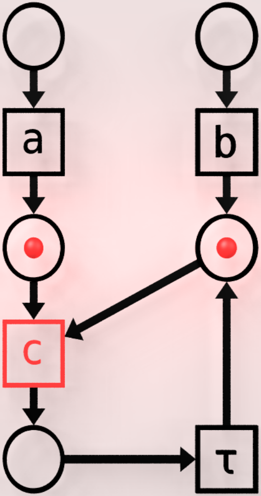
- A multiset $M \in \mathbb{N}^S$ is a *marking* of N.
- $t \in T$ is *enabled* if ${}^\bullet t \leq M$.
- A nonempty, finite $G \in \mathbb{N}^T$ is a *step* from M to marking M' iff ${}^\bullet G \leq M$ and $M' = M - {}^\bullet G + G^\bullet$.
- $[M_0\rangle$ denotes the set of *reachable* markings.
- If $[M_0\rangle \subseteq \{0, 1\}^S$ the net is *1-safe*.



Dynamic Behaviour – Firing Rule

Let $N = (S, T, F, M_0, \ell)$ be a net.

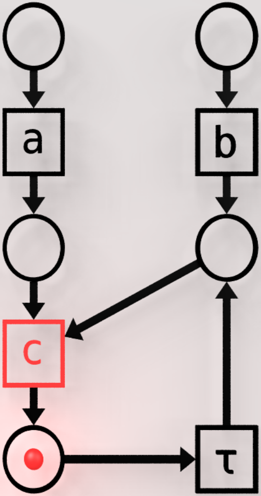
- A multiset $M \in \mathbb{N}^S$ is a *marking* of N.
- $t \in T$ is *enabled* if ${}^\bullet t \leq M$.
- A nonempty, finite $G \in \mathbb{N}^T$ is a *step* from M to marking M' iff ${}^\bullet G \leq M$ and $M' = M - {}^\bullet G + G^\bullet$.
- $[M_0\rangle$ denotes the set of *reachable* markings.
- If $[M_0\rangle \subseteq \{0, 1\}^S$ the net is *1-safe*.



Dynamic Behaviour – Firing Rule

Let $N = (S, T, F, M_0, \ell)$ be a net.

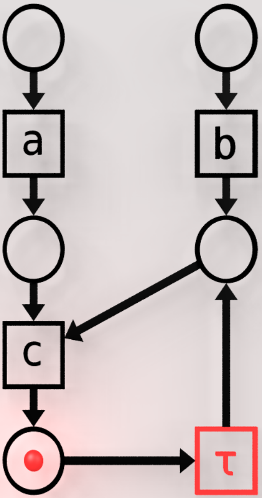
- A multiset $M \in \mathbb{N}^S$ is a *marking* of N.
- $t \in T$ is *enabled* if ${}^\bullet t \leq M$.
- A nonempty, finite $G \in \mathbb{N}^T$ is a *step* from M to marking M' iff ${}^\bullet G \leq M$ and $M' = M - {}^\bullet G + G^\bullet$.
- $[M_0\rangle$ denotes the set of *reachable* markings.
- If $[M_0\rangle \subseteq \{0, 1\}^S$ the net is *1-safe*.



Dynamic Behaviour – Firing Rule

Let $N = (S, T, F, M_0, \ell)$ be a net.

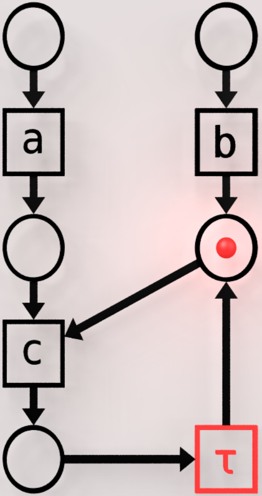
- A multiset $M \in \mathbb{N}^S$ is a *marking* of N.
- $t \in T$ is *enabled* if $^\bullet t \leq M$.
- A nonempty, finite $G \in \mathbb{N}^T$ is a *step* from M to marking M' iff $^\bullet G \leq M$ and $M' = M - {}^\bullet G + G^\bullet$.
- $[M_0\rangle$ denotes the set of *reachable* markings.
- If $[M_0\rangle \subseteq \{0, 1\}^S$ the net is *1-safe*.



Dynamic Behaviour – Firing Rule

Let $N = (S, T, F, M_0, \ell)$ be a net.

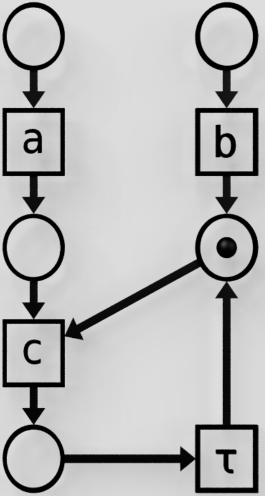
- A multiset $M \in \mathbb{N}^S$ is a *marking* of N.
- $t \in T$ is *enabled* if ${}^\bullet t \leq M$.
- A nonempty, finite $G \in \mathbb{N}^T$ is a *step* from M to marking M' iff ${}^\bullet G \leq M$ and $M' = M - {}^\bullet G + G^\bullet$.
- $[M_0\rangle$ denotes the set of *reachable* markings.
- If $[M_0\rangle \subseteq \{0, 1\}^S$ the net is *1-safe*.



Dynamic Behaviour – Firing Rule

Let $N = (S, T, F, M_0, \ell)$ be a net.

- A multiset $M \in \mathbb{N}^S$ is a *marking* of N.
- $t \in T$ is *enabled* if ${}^\bullet t \leq M$.
- A nonempty, finite $G \in \mathbb{N}^T$ is a *step* from M to marking M' iff ${}^\bullet G \leq M$ and $M' = M - {}^\bullet G + G^\bullet$.
- $[M_0\rangle$ denotes the set of *reachable* markings.
- If $[M_0\rangle \subseteq \{0, 1\}^S$ the net is *1-safe*.



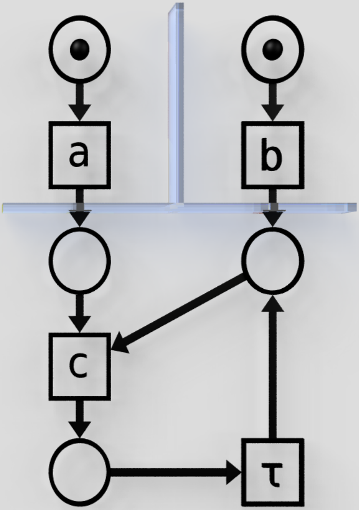
Distributed Nets

Let $N = (S, T, F, M_0, \ell)$ be a net.

An equivalence relation $\equiv_D \subseteq (S \cup T) \times (S \cup T)$ is a *distribution* iff

- $\forall t \in T, s \in {}^\bullet t. s \equiv_D t$, and
- if $M \in [M_0\rangle$ and $M[\{t, u\}\rangle M'$ then $s \not\equiv_D t$.

N is *distributed* if any distribution exists.



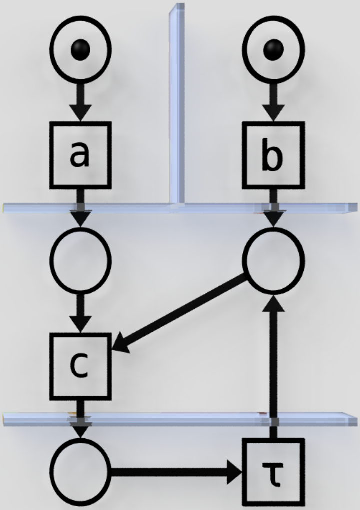
Distributed Nets

Let $N = (S, T, F, M_0, \ell)$ be a net.

An equivalence relation $\equiv_D \subseteq (S \cup T) \times (S \cup T)$ is a *distribution* iff

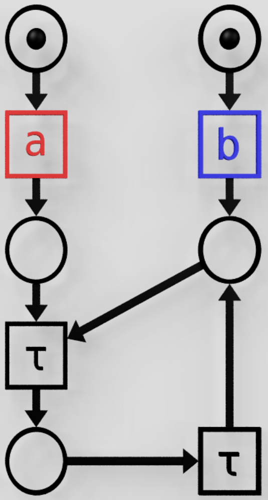
- $\forall t \in T, s \in {}^\bullet t . s \equiv_D t$, and
- if $M \in [M_0\rangle$ and $M[\{t, u\}\rangle M'$ then $s \not\equiv_D t$.

N is *distributed* if any distribution exists.



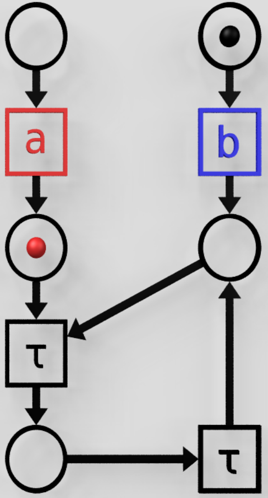
Firing Rule with Causality

- Non-τ transitions denote interactions with the environment.
- Tokens remember causal history.
- When recording only set of historic labels, statespace becomes finite for finite nets.



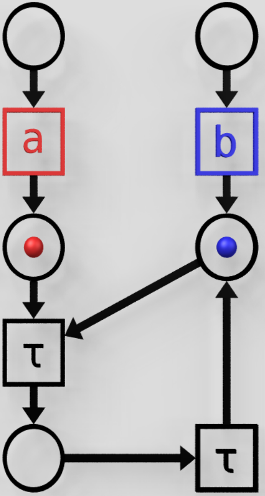
Firing Rule with Causality

- Non-τ transitions denote interactions with the environment.
- Tokens remember causal history.
- When recording only set of historic labels, statespace becomes finite for finite nets.



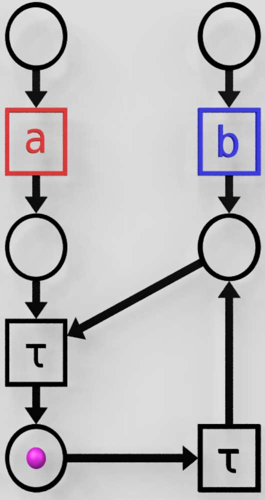
Firing Rule with Causality

- Non-τ transitions denote interactions with the environment.
- Tokens remember causal history.
- When recording only set of historic labels, statespace becomes finite for finite nets.



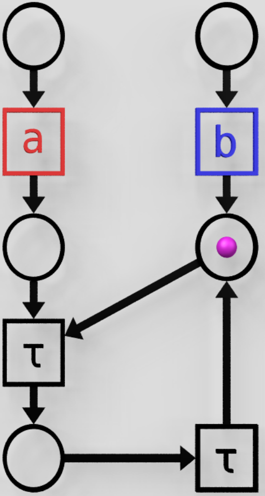
Firing Rule with Causality

- Non-τ transitions denote interactions with the environment.
- Tokens remember causal history.
- When recording only set of historic labels, statespace becomes finite for finite nets.



Firing Rule with Causality

- Non-τ transitions denote interactions with the environment.
- Tokens remember causal history.
- When recording only set of historic labels, statespace becomes finite for finite nets.



Processes

A pair $\mathcal{P} = (\mathcal{N}, \pi)$ is a *process* of a net $N = (S, T, F, M_0, \ell)$ iff

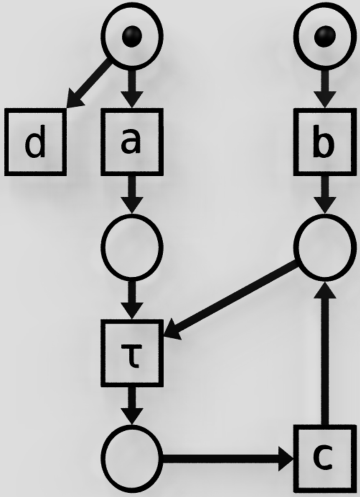
- $\mathcal{N} = (\mathcal{S}, \mathcal{T}, \mathcal{F}, \mathcal{M}_0, \mathcal{l})$ is a net, satisfying
  - $\forall s \in \mathcal{S}. |{}^\bullet s| \leq 1 \geq |s^\bullet| \wedge \mathcal{M}_0(s) = \begin{cases} 1 & \text{iff } {}^\bullet s = \emptyset \\ 0 & \text{otherwise} \end{cases},$
  - all arc-weights are 1, i. e. $\mathcal{F}(x, y) \in \{0, 1\}$ for all x, y and $\mathcal{F}$ can be considered a relation,
  - $\mathcal{F}$ is acyclic, i. e. $\forall x \in \mathcal{S} \cup \mathcal{T}. (x, x) \notin \mathcal{F}^+$, where $\mathcal{F}^+$ is the transitive closure of $\mathcal{F}$,
  - and $\{t \mid (t, u) \in \mathcal{F}^+\}$ is finite for all $u \in \mathcal{T}$.
- $\pi : \mathcal{S} \cup \mathcal{T} \rightarrow S \cup T$ is a function with $\pi(\mathcal{S}) \subseteq S$ and $\pi(\mathcal{T}) \subseteq T$, satisfying
  - $|\pi^{-1}(s) \cap \mathcal{M}_0| = M_0(s)$ for all $s \in S$,
  - $\forall t \in \mathcal{T}, s \in S.\ F(s, \pi(t)) = |\pi^{-1}(s) \cap {}^\bullet t| \wedge F(\pi(t), s) = |\pi^{-1}(s) \cap t^\bullet|$, and
  - $\forall t \in \mathcal{T}. \mathcal{l}(t) = \ell(\pi(t))$.

Processes

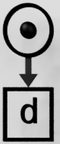
Let $\mathcal{P} = (\mathcal{N}, \pi)$ be a process and $\mathcal{N} = (\mathcal{S}, \mathcal{T}, \mathcal{F}, \mathcal{M}_0, \ell)$.

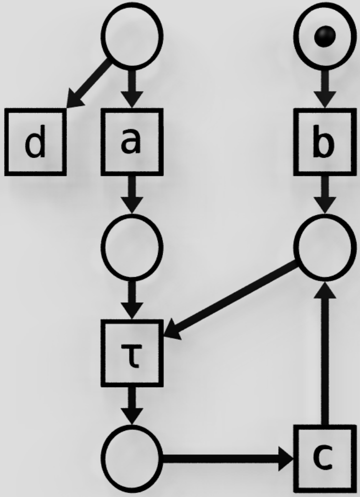
- The *end of the net* $\mathcal{N}^\circ$ is the set $\{s \in \mathcal{S} \mid s^\bullet = \emptyset\}$.
- $\mathcal{P}$ is *maximal* iff $\nexists G . \pi(\mathcal{N}^\circ)[G\rangle_N$.
- The set of all maximal processes of a net N is denoted by $MP(N)$.

Example: Processes of a Petri Net

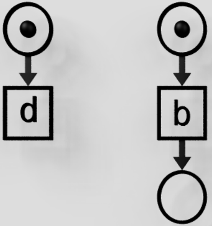


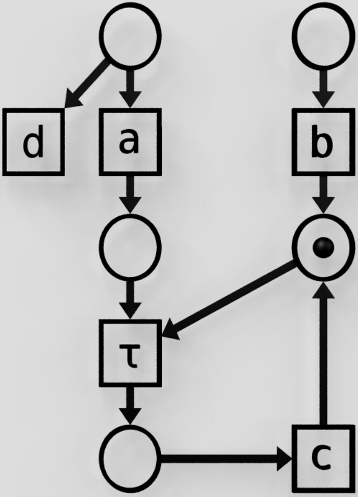
Example: Processes of a Petri Net



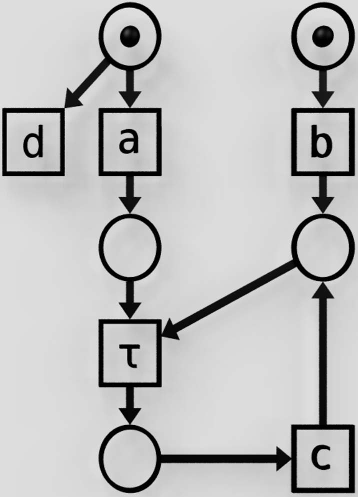


Example: Processes of a Petri Net

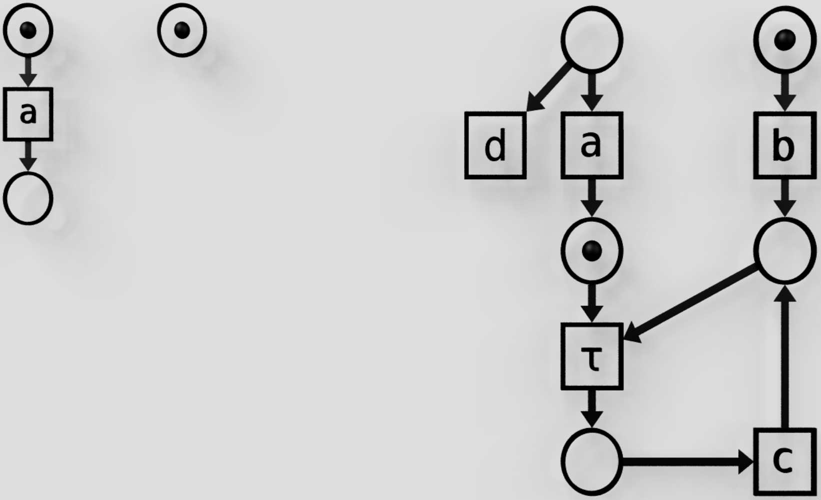




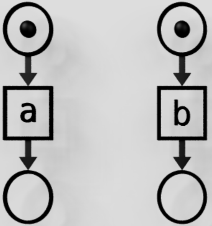
Example: Processes of a Petri Net

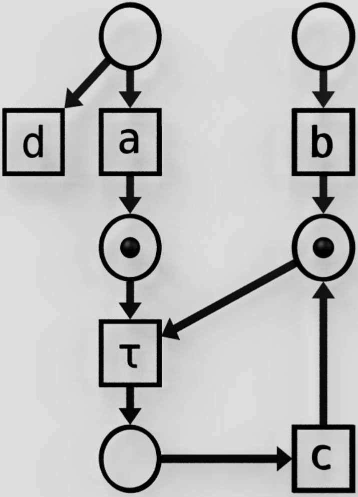


Example: Processes of a Petri Net

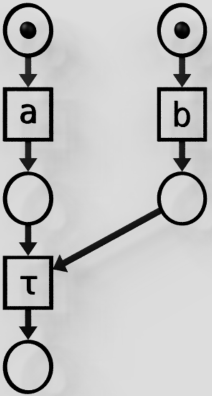


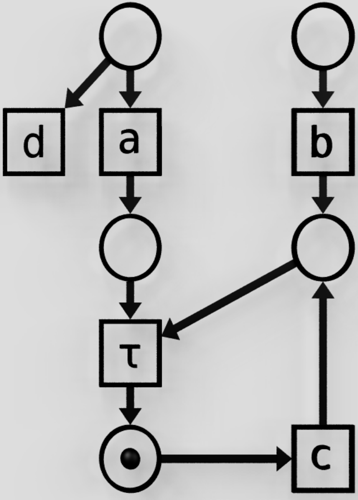
Example: Processes of a Petri Net



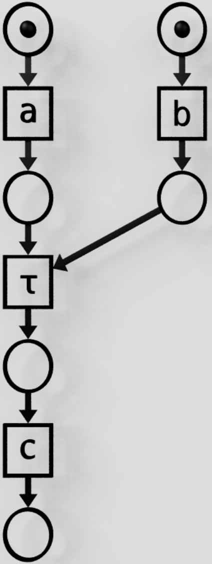


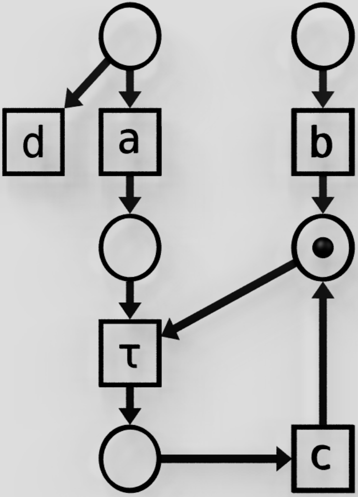
Example: Processes of a Petri Net



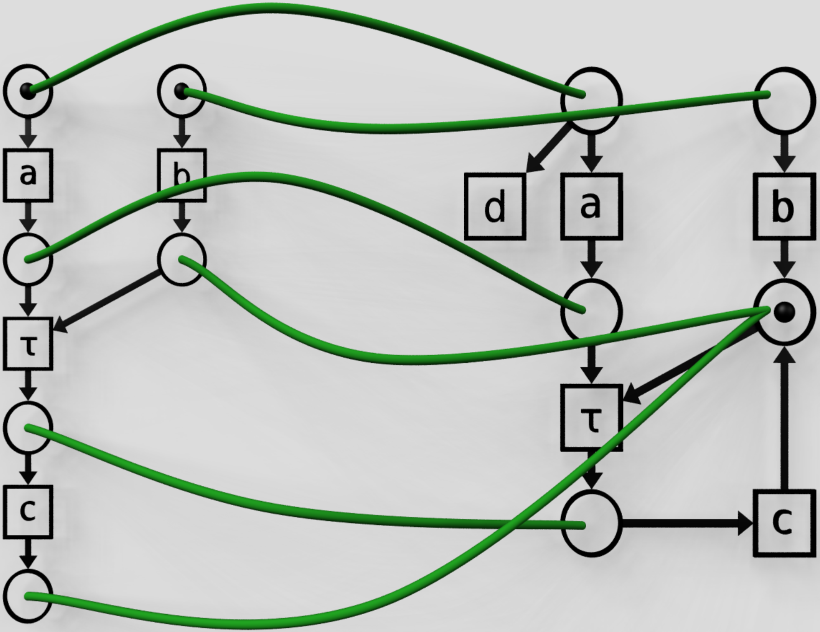


Example: Processes of a Petri Net





Example: Processes of a Petri Net



Labelled Partial Orders

A *labelled partial order* is a structure $(V, T, \leq, \ell)$ where

- V is a set of *vertices*,
- T is a set of *labels*,
- $\leq \subseteq V \times V$ is a partial order relation,
- $\ell : V \to T$ (the *labelling* function).

Pomsets

Two labelled partial orders $o = (V, T, \leq, \ell)$ and $o' = (V', T', \leq', \ell')$ are *isomorphic*, $o \approxeq o'$ iff there exists a bijection $\phi : V \to V'$ such that

- $\forall v \in V . \ell(v) = \ell'(\phi(v))$ and
- $\forall u, v \in V . u \leq v \Leftrightarrow \phi(u) \leq \phi(v)$.

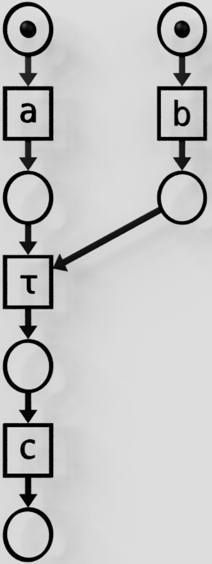
The *pomset* of o is its isomorphism class $[o] := \{o' \mid o \approxeq o'\}$.

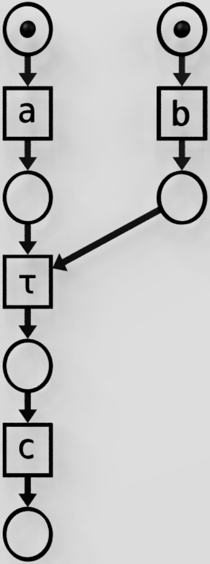
Pomset Traces

Let $\mathcal{P} = ((\mathcal{S}, \mathcal{T}, \mathcal{F}, \mathcal{M}_0, \ell), \pi)$ be a process.

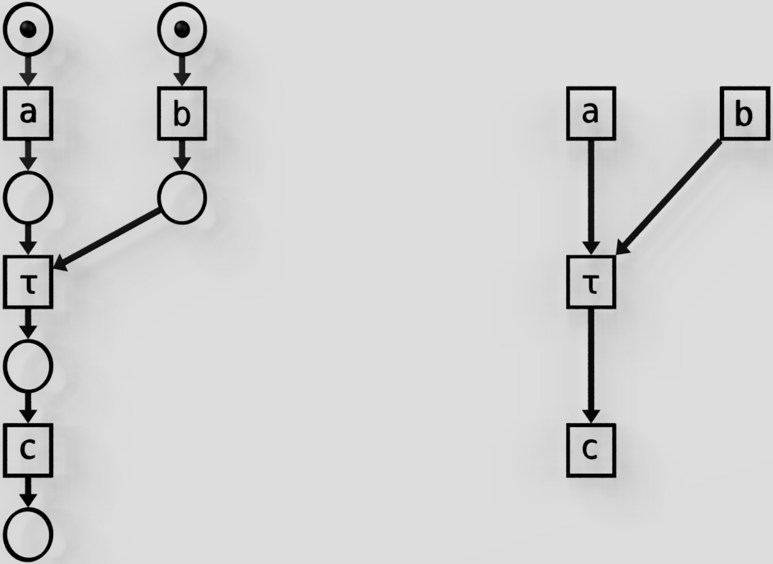
- Let $\mathcal{O} := \{t \in \mathcal{T} \mid \ell(t) \neq \tau\}$, i.e. the visible transitions of the process.
- The *visible pomset* of $\mathcal{P}$ is the pomset $VP(\mathcal{P}) := [(\mathcal{O}, \mathrm{Act}, \mathcal{F}^* \cap \mathcal{O} \times \mathcal{O}, \ell \cap (\mathcal{O} \times \mathrm{Act}))]$ where $\mathcal{F}^*$ is the transitive and reflexive closure of the flow relation $\mathcal{F}$.
- $\mathrm{MVP}(N) := \{VP(\mathcal{P}) \mid \mathcal{P} \in MP(N)\}$ is the set of visible pomsets of all maximal processes of N.
- Two nets N and N' are *completed pomset trace equivalent*, $N \approx_{\mathcal{C}PT} N'$, iff $\mathrm{MVP}(N) = \mathrm{MVP}(N')$.

Example: Completed Pomset Trace

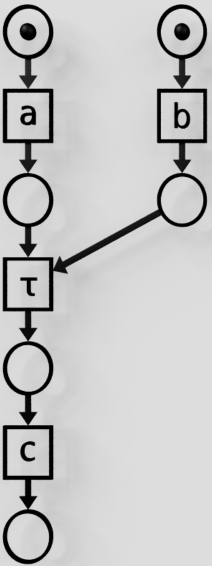




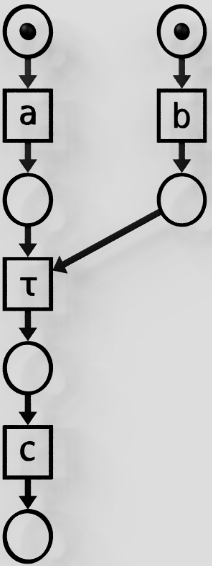
Example: Completed Pomset Trace



Example: Completed Pomset Trace



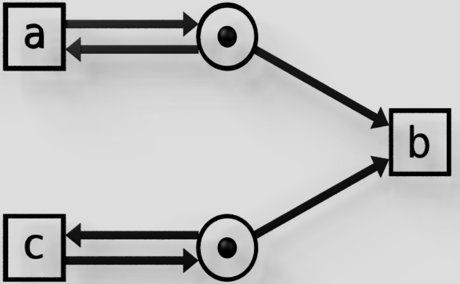
Example: Completed Pomset Trace



Completed Pomset Trace Equivalence

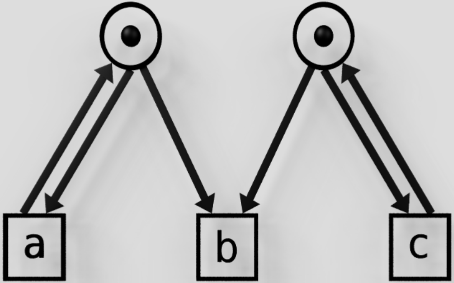
- Tracks causality
- Tracks deadlocks
- Tracks divergence
- Abstracts from transition identities
- Abstracts from decision structure
- Abstracts from non-diverging silent transitions

Formal Problem Statement



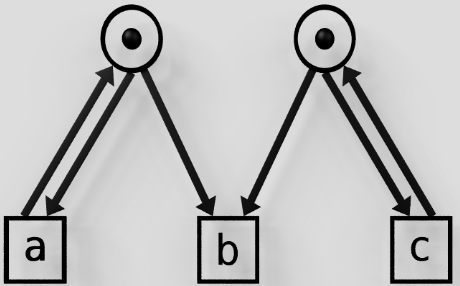
Can we find a 1-safe, finite, and distributed Petri net which is completed pomset trace equivalent to this specification net?

Formal Problem Statement



Can we find a 1-safe, finite, and distributed Petri net which is completed pomset trace equivalent to this specification net?

Formal Problem Statement



Can we find a 1-safe, finite, and distributed Petri net which is completed pomset trace equivalent to this specification net?

No.

Sketch of the Proof

- Track only token colour, not full history.
- Extend markings to *dependency markings* $M : (S \times 2^{\mathrm{Act}}) \to \mathbb{N}$.
- Finite 1-safe net has infinite runs, but statespace of size at most $m := (1 + 2^{|\mathrm{Act}|})^{|S|}$, i.e. finite.

Lemma: For a dependency marking M,
if $M[\{t_1\}\rangle[\{t_2\}\rangle \cdots [\{t_n\}\rangle M$

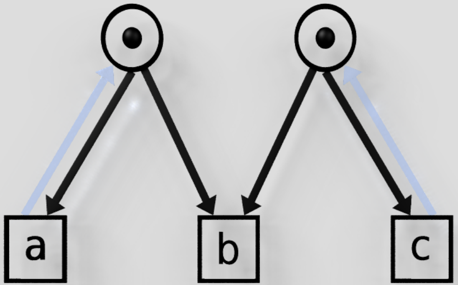
- all tokens produced by t_i have the same dependencies as those consumed,
- as otherwise, the less-dependent tokens could have been produced without t_i; violating 1-safety.

Sketch of the Proof

Theorem: There is no 1-safe, finite, distributed Petri net which is completed pomset trace equivalent to our specification.

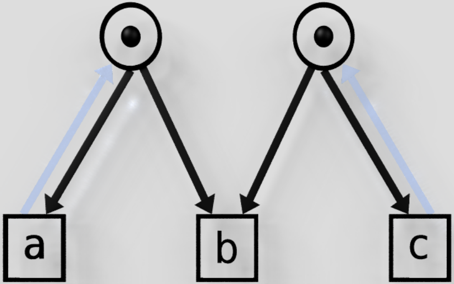
- Specification can fire $(ac)^m b$.
- While doing so, some dependency marking M_l must be reached twice.
- With the Lemma, partition the loop into *a*-coloured and *c*-coloured part.
- While a^m can be fired, must also be able to fire c^m, otherwise new pomset with finitely many c but infinitely many a is generated. Dito with a and c reversed.
- In $(ac)^m b$ a single transition must have consumed an *a*-coloured and a *c*-coloured token, hence these two tokens reside on co-located places.
- As these tokens lead independently to a^m resp. c^m there are two concurrently firing transactions consuming them, hence they must be on different locations.

Core of the Problem



The “M” (i.e. optional coordination).

Core of the Problem



The “M” (i.e. optional coordination).

Is this particular to completed pomset trace equivalence?

Finite Step Failures Equivalence

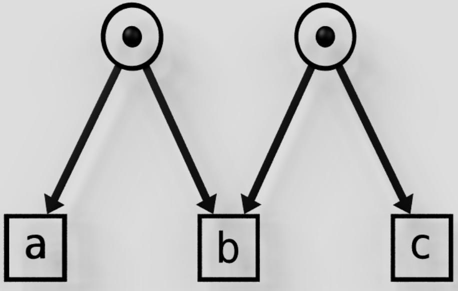
Whenever the net can only fire visible transitions, record a step failure pair, i. e.

- the trace of labels leading up to this marking, and
- all finite multisets of labels which can not fire in the next step.

Compare set of recorded step failure pairs.

- Abstracts from causality
- Tracks deadlocks
- Tracks divergence
- Abstracts from transition identities
- Tracks decision structure
- Abstracts from non-diverging silent transitions
- Tracks concurrency

Counterexample for Finite Step Failures Equivalence



Coarser than Finite Step Failures?

Without branching structure (“linear–time”):

- Decide everything on central location, execute visible transitions on distributed locations

With interleaving semantics:

- Connect all transitions to central scheduling place

When allowing divergence:

- Busy-wait all decisions

Finer than Finite Step Failures?

- Results stable up to branching ST-bisimilarity with explicit divergence.
- Includes practically the entire branching-time part of Rob's spectrum.

Structures Smaller than M?

- Answer comprises the largest part of my thesis
- . . . because everything non-M can be implemented

Structures Smaller than M?

- Answer comprises the largest part of my thesis
- ... because everything non-M can be implemented
- About 10 pages to describe the necessary Petri net construction
- About 30 pages for the correctness proof

Structures Smaller than M?

- Answer comprises the largest part of my thesis
- . . . because everything non-M can be implemented
- About 10 pages to describe the necessary Petri net construction
- About 30 pages for the correctness proof (with huge invariant)

Structures Smaller than M?

- Answer comprises the largest part of my thesis
- . . . because everything non-M can be implemented
- About 10 pages to describe the necessary Petri net construction
- About 30 pages for the correctness proof (with huge invariant)
- About 8 pages to describe a Petri net to distributed C compiler used for testing

Structures Smaller than M?

- Answer comprises the largest part of my thesis
- ...because everything non-M can be implemented
- About 10 pages to describe the necessary Petri net construction
- About 30 pages for the correctness proof (with huge invariant)
- About 8 pages to describe a Petri net to distributed C compiler used for testing
- ...and finding a bug in the already published construction and proof

What About Other Formal Models?

How dependent are the results on the choice of Petri nets specifically?

- Not terribly so: π-calculus results co-developed by Peters & Nestmann, formal connection was being worked on by Mennicke.
- Some hardware-centric results by Lamport seem related, no formal connection established.

Ways to Evade the Negative Theorems

- Don't use branching time and solve the consensus problems probabilistically.
- Assume bounded message delays (often needed for error detection anyway).
- Use approximately uniform passage of time.
- Physical effects not accurately captured by Petri nets.

Open Problems and Questions

- Where between weak completed step trace equivalence and finite step failures equivalence become Ms unimplementable?
- Conjecture: There is some “asynchronous branching time” equivalence (and Ms are implementable therein).
- Which structure(s) delineate(s) the limit of distributed implementability when checking all three of divergence, causality and branching time?
- Stability of Ms in non-safe nets under causality only conjectured so far.
- Efficient modelling of quantum-mechanical effects for distributed computing.

References and Earlier Publications

The thesis includes content from various papers with Glabbeek, Goltz, Mennicke, Nestmann, Peters (alphabetically ordered).

“External” must-reads:

- Best and Darondeau: “Petri net distributability”
- Gorla: “On the relative expressive power of asynchronous communication primitives”
- Hopkins: “Distributable nets”
- Palamidessi: “Comparing the expressive power of the synchronous and the asynchronous π-calculus”
- Taubner: “Zur verteilten Implementierung von Petrinetzen”

Results

- Identified the M as a problematic structure for distributed implementations
- Showed stability under causality respecting equivalences
- Showed stability under branching time equivalences
- Showed the M to be the smallest such structure, by concrete implementation for all other cases
- Showed an infinite hierarchy of bigger problematic structures exists
- Established formal connections between free-choice Petri nets and asynchronous nets (omitted in this talk)
- Described LSGA-nets as an alternative and equivalent approach to generate distributed nets (dito)
- Described structural conflict nets and showed them to be a class of nets the implementation is valid for (dito)

Thank You!

- Institut für Programmierung und Reaktive Systeme @ TU Braunschweig
- National ICT Australia
- Deutsche Forschungsgemeinschaft
- Deutscher Akademischer Austauschdienst
- Studienstiftung des deutschen Volkes