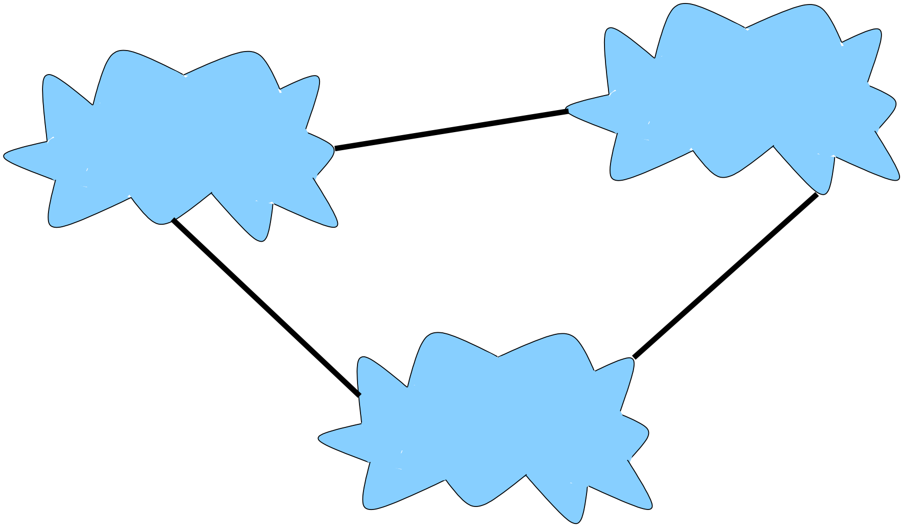
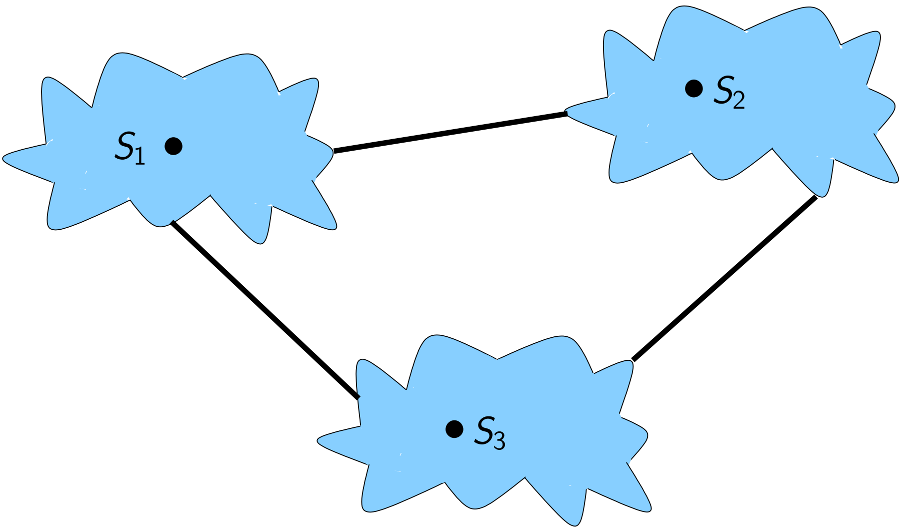
Monotony in Service Orchestrations

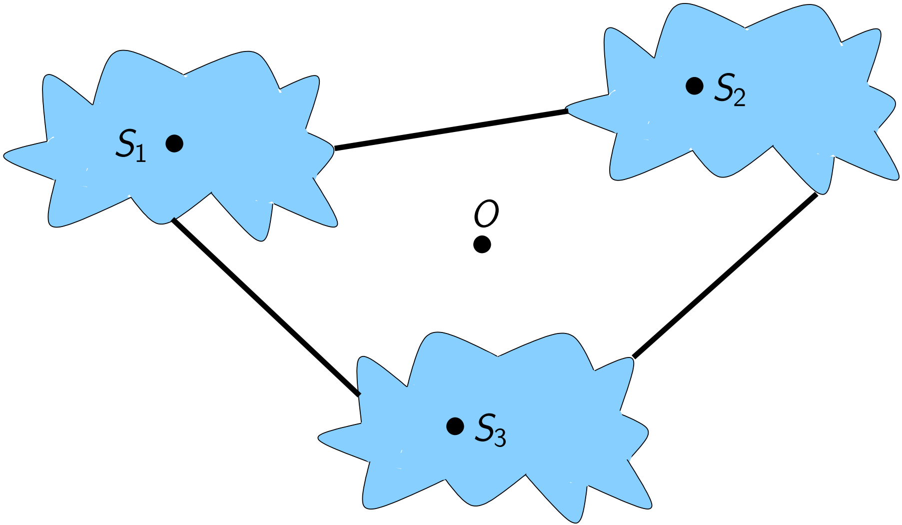
A. Bouillard S. Rosario A. Benveniste S. Haar

Web Services and Orchestrations



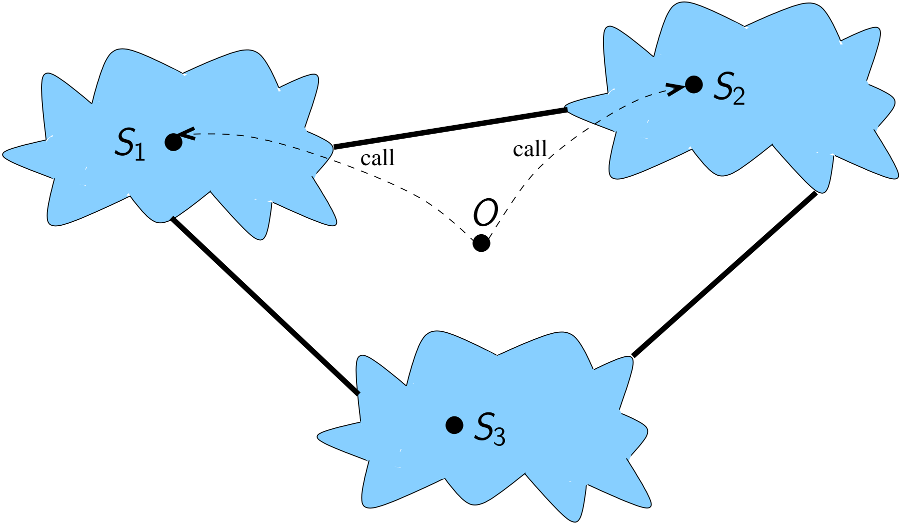
Web Services and Orchestrations

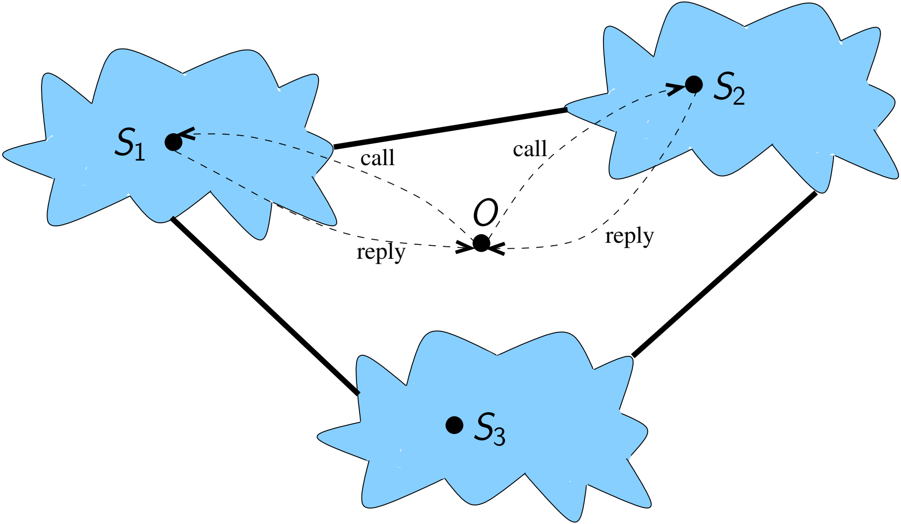




S_1
S_2
O
S_3

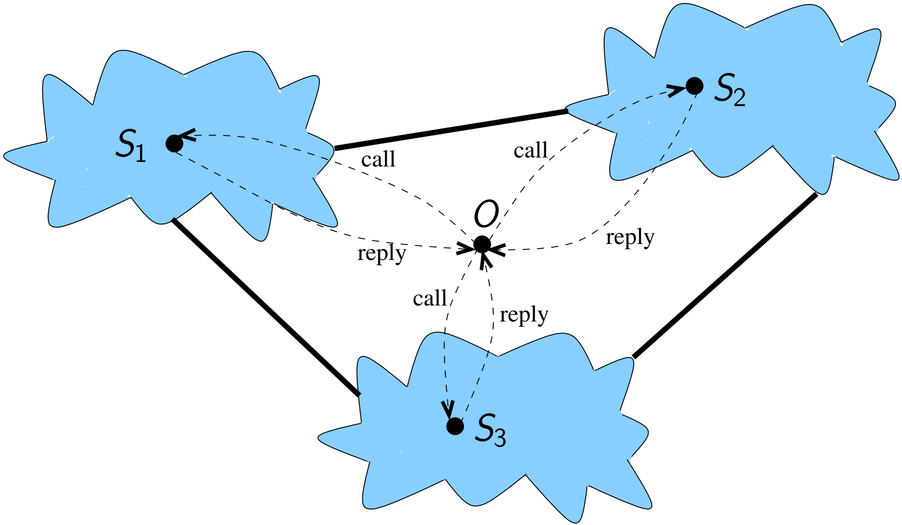
Web Services and Orchestrations



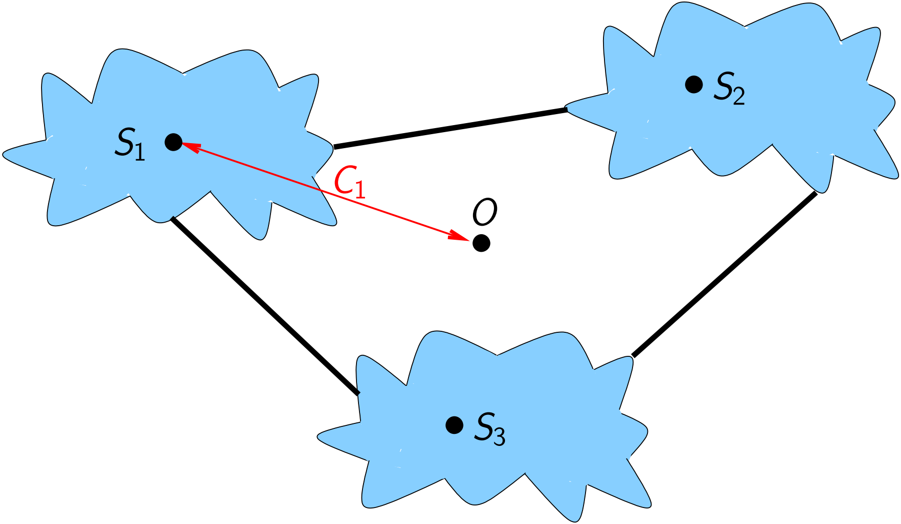


S_1
S_2
S_3
O
call
call
reply
reply

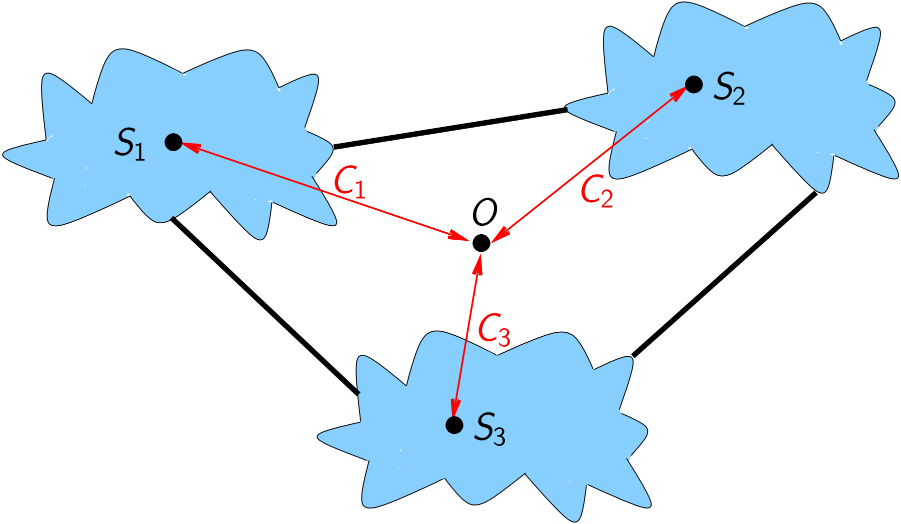
Web Services and Orchestrations



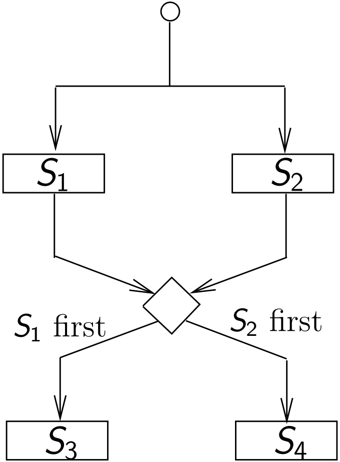
Contract, Contract Composition



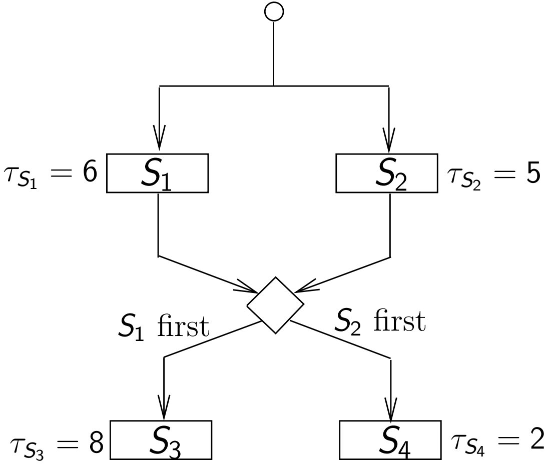
Contract, Contract Composition



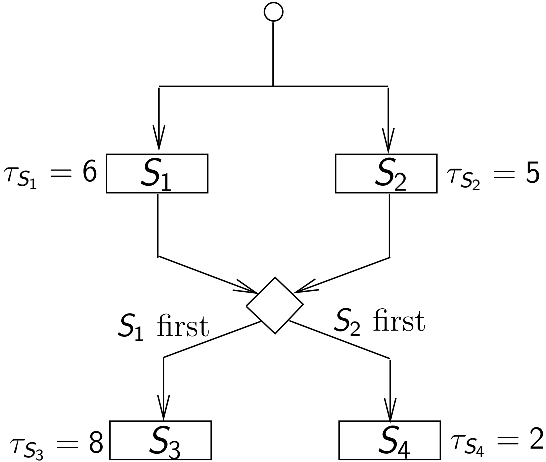
A non-monotonic orchestration



A non-monotonic orchestration

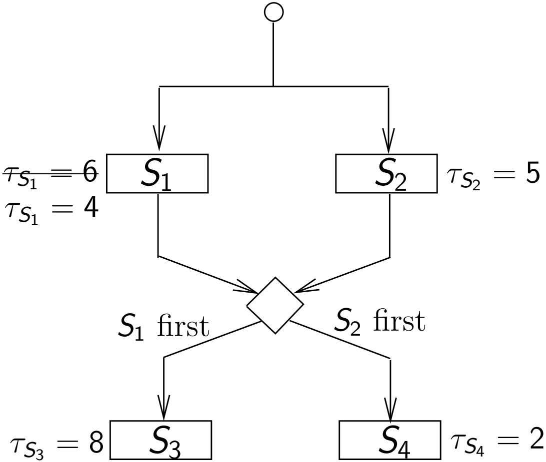


A non-monotonic orchestration

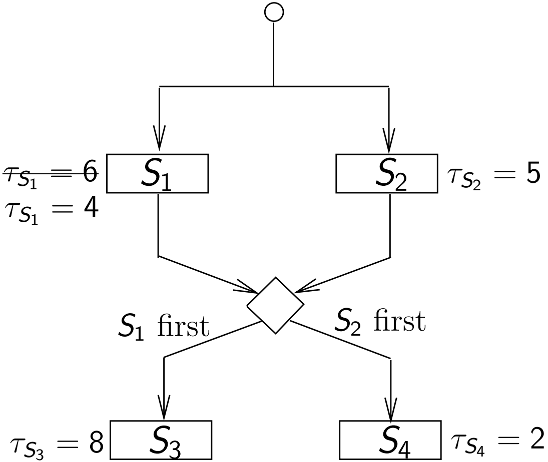


Overall latency : $\tau_{S_2} + \tau_{S_4} = 7$

A non-monotonic orchestration



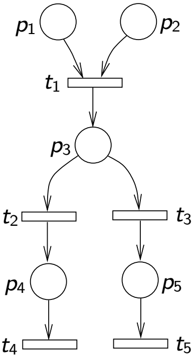
A non-monotonic orchestration



Overall latency : $\tau_{S_1} + \tau_{S_3} = 12$

Colored, Occurrence Nets: to model orchestrations

Petri Nets



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

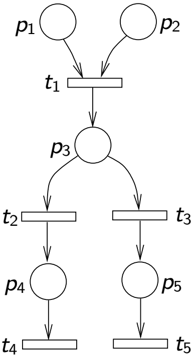
$\mathcal{P}$: Set of Places

$\mathcal{T}$: Set of Transitions

$\mathcal{F} \subseteq (\mathcal{P} \times \mathcal{T}) \cup (\mathcal{T} \times \mathcal{P})$: Flow Relation

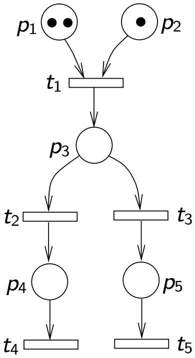
$M_0 : \mathcal{P} \rightarrow \mathbb{N}$: Initial Marking

Petri Nets



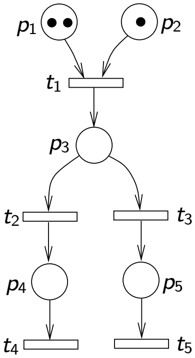
$$M_0 = \{p_1 \rightarrow 2, p_2 \rightarrow 1, p_3 \rightarrow 0$$
$$p_4 \rightarrow 0, p_5 \rightarrow 0\}$$

Petri Nets



$$M_0 = \{p_1 \rightarrow 2, p_2 \rightarrow 1, p_3 \rightarrow 0$$
$$p_4 \rightarrow 0, p_5 \rightarrow 0\}$$

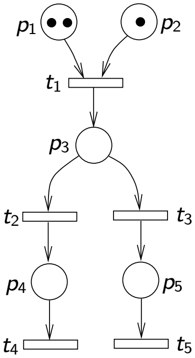
Petri Nets



Preset of a node x : $\bullet x$

for *e.g*, $\bullet t_1 = \{p_1, p_2\}$

Petri Nets



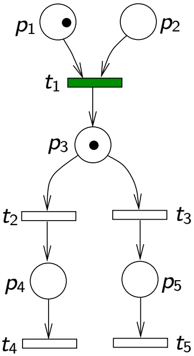
Preset of a node x : $^\bullet x$

for $e.g$, $^\bullet t_1 = \{p_1, p_2\}$

Postset of a node x : $x^\bullet$

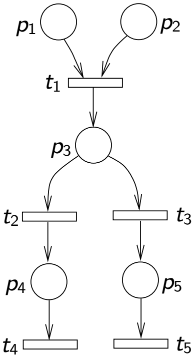
for $e.g$, $t_1{}^\bullet = \{p_3\}$

Petri Nets



Firing of transition t_1

Petri Nets

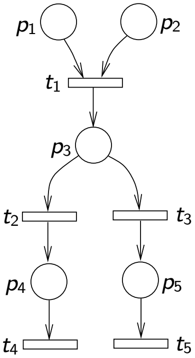


Causality Relation : $\leq$

for e.g, : $p_3 < t_1 < p_1$

$t_1 < t_2 < t_4$

Petri Nets



Causality Relation : $\leq$

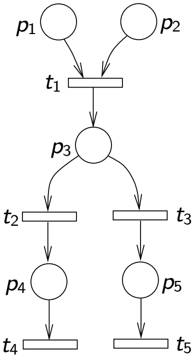
for e.g, : $p_3 < t_1 < p_1$

$t_1 < t_2 < t_4$

Conflict Relation : $\#$

for e.g, : $t_2 \# t_3$

Petri Nets



Causality Relation : $\leq$

for e.g, : $p_3 < t_1 < p_1$

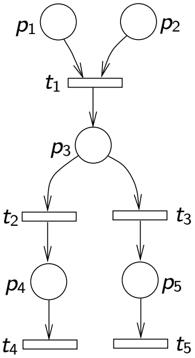
$t_1 < t_2 < t_4$

Conflict Relation : $\#$

for e.g, : $t_2 \# t_3$

$t_4 \# t_5$

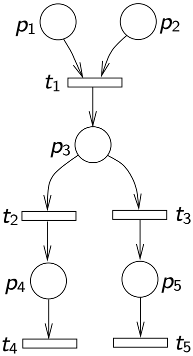
Petri Nets



A **configuration** is a sub-net κ s.t.:

1. κ is causally closed.
   If $x \in \kappa$ and $x' < x$, then $x' \in \kappa$.

Petri Nets



A **configuration** is a sub-net κ s.t.:

1. κ is causally closed.
   If $x \in \kappa$ and $x' < x$, then $x' \in \kappa$.

2. κ is conflict-free.
   $\nexists x, x' \in \kappa \; s.t. \; x \# x'$

Occurrence Nets

A safe net N is called an *occurrence net* iff

1. No node of N is in self-conflict.
2. $\leq$ is a partial order
3. $\lceil t \rceil = \{x \in N \mid x \leq t\}$ is finite for all transitions of N.
4. $|{}^{\bullet}p| \leq 1$ for all places of N.

Our Model: **OrchNets**

OrchNets

- Tokens have *colors*: (value, date)

OrchNets

- Tokens have *colors*: (value, date)
- A transition t has functions (ϕ_t, τ_t) that modify the token colors.

OrchNets: Definition

An *OrchNet* is a tuple $\mathcal{N} = (N, \Phi, T, T_{\mathrm{init}})$

OrchNets: Definition

An *OrchNet* is a tuple $\mathcal{N} = (N, \Phi, T, T_{\text{init}})$

- N : occurrence net with token attributes $c = (value, date)$.

OrchNets: Definition

An *OrchNet* is a tuple $\mathcal{N} = (N, \Phi, T, T_{\text{init}})$

- N : occurrence net with token attributes $c = (value, date)$.
- $\Phi = (\phi_t)_{t \in \mathcal{T}}$: family of *value functions*.

OrchNets: Definition

An *OrchNet* is a tuple $\mathcal{N} = (N, \Phi, T, T_{\mathrm{init}})$

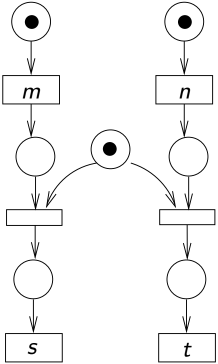
- N : occurrence net with token attributes $c = (value, date)$.
- $\Phi = (\phi_t)_{t \in \mathcal{T}}$: family of *value functions.*
- $T = (\tau_t)_{t \in \mathcal{T}}$: family of *latency functions.*

OrchNets: Definition

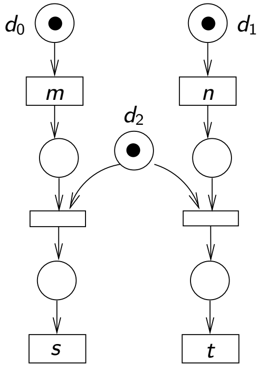
An *OrchNet* is a tuple $\mathcal{N} = (N, \Phi, T, T_{\text{init}})$

- N : occurrence net with token attributes $c = (\mathit{value}, \mathit{date})$.
- $\Phi = (\phi_t)_{t \in \mathcal{T}}$: family of *value functions.*
- $T = (\tau_t)_{t \in \mathcal{T}}$: family of *latency functions.*
- $T_{\text{init}} = (\tau_p)_{p \in \min(\mathcal{P})}$: family of *initial date functions*.

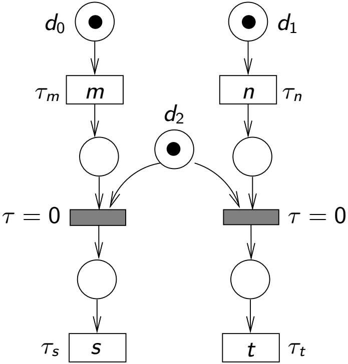
OrchNets: Example



OrchNets: Example



OrchNets: Example



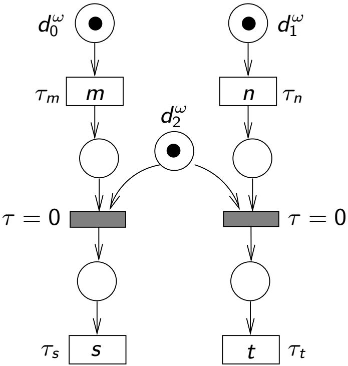
In general, ϕ_t and τ_t are non-deterministic functions.

$\omega \in \Omega$, a **daemon** variable that resolves non-determinism.

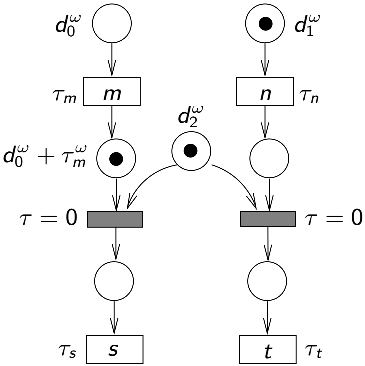
$\omega \in \Omega$, a **daemon** variable that resolves non-determinism.

For a given value of ω, ϕ_t^ω and τ_t^ω are deterministic functions.

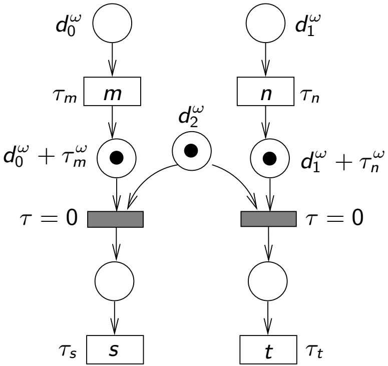
For a fixed ω



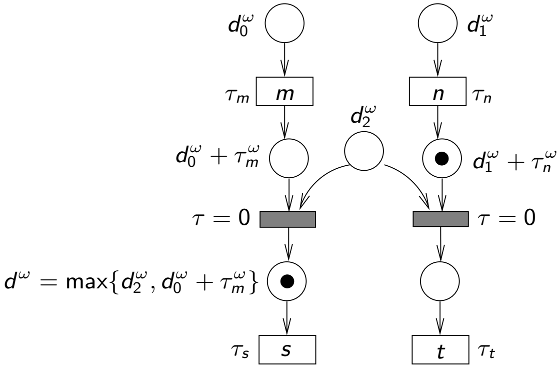
For a fixed ω



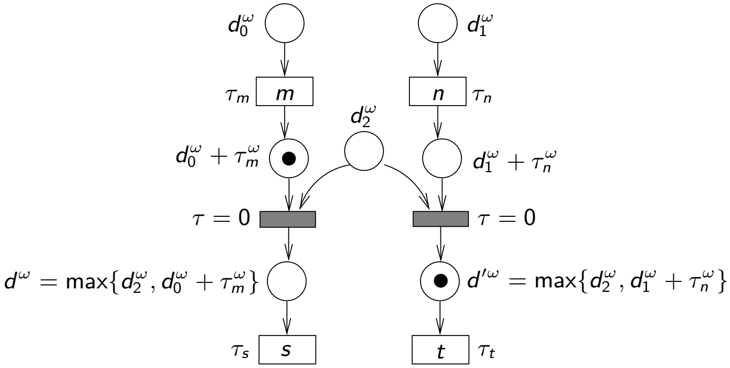
For a fixed ω



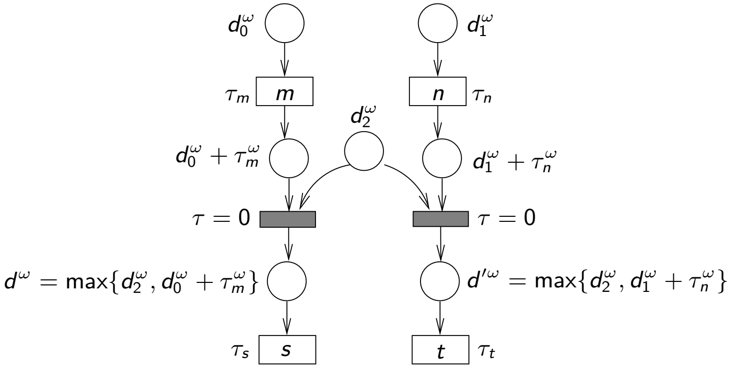
For a fixed ω



For a fixed ω



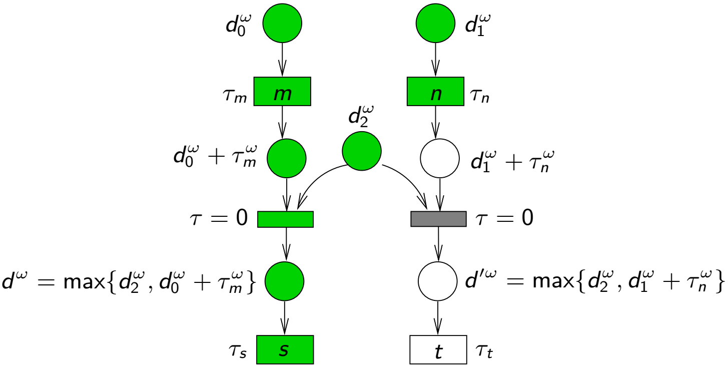
For a fixed ω



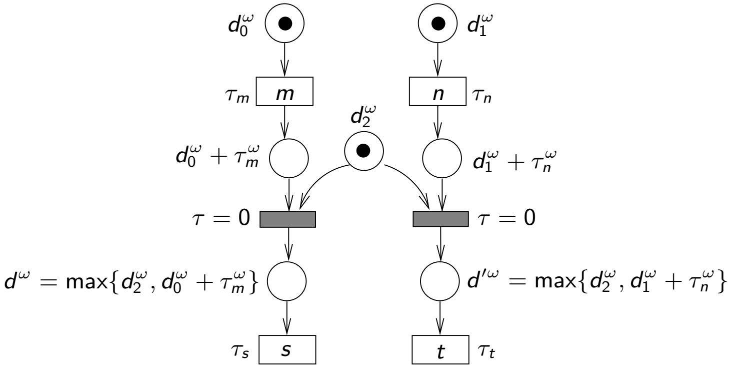
Actually occurring configuration for a given ω: $\overline{\kappa}(\mathcal{N}, \omega)$

When $d^\omega < d'^\omega$, $\overline{\kappa}(\mathcal{N}, \omega)$ is..

When $d^\omega < d'^\omega$, $\overline{\kappa}(\mathcal{N}, \omega)$ is..



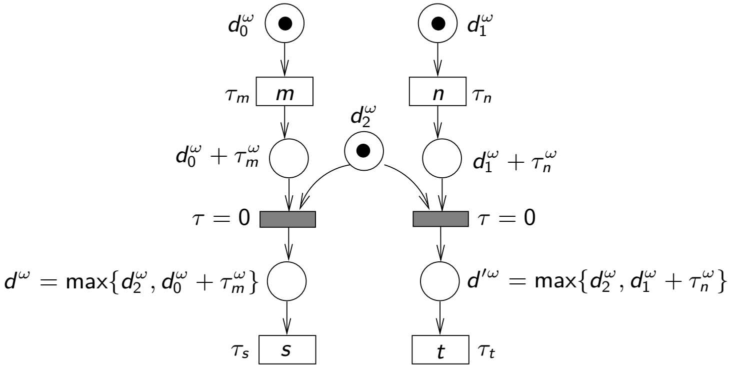
Execution Time



Execution Time of a maximal configuration $\overline{\kappa}$ of $\mathcal{N}$:

$$E_\omega(\overline{\kappa}, \mathcal{N}) = \{max(d_x^\omega) \mid x \in \overline{\kappa}\}$$

Execution Time

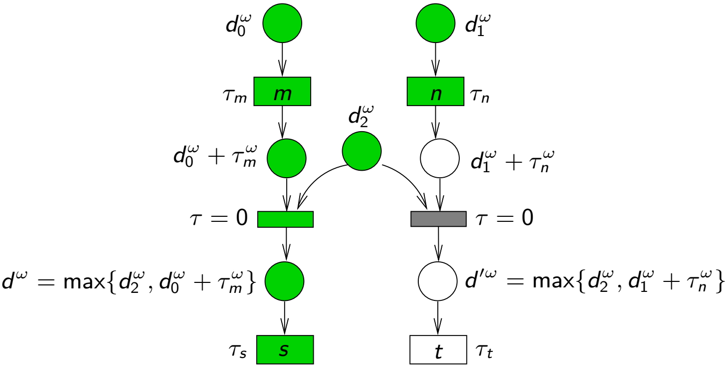


For a given ω, **execution time** of $\mathcal{N}$:

$$E_\omega(\mathcal{N}) = E_\omega(\overline{\kappa}(\mathcal{N}, \omega), \mathcal{N})$$

When $d^\omega < d'^\omega$..

When $d^\omega < d'^\omega$..



$$E_\omega(\mathcal{N}) = d^\omega + \tau_s^\omega$$

Characterising Monotony..

Pre-Orchnets

Call **pre-OrchNet** a tuple $\mathbb{N} = (N, \Phi, \mathbb{T}, \mathbb{T}_{\text{init}})$ where,

Pre-Orchnets

Call **pre-OrchNet** a tuple $\mathbb{N} = (N, \Phi, \mathbb{T}, \mathbb{T}_{\text{init}})$ where,

1. N, Φ : as before.

Pre-Orchnets

Call **pre-OrchNet** a tuple $\mathbb{N} = (N, \Phi, \mathbb{T}, \mathbb{T}_{\mathrm{init}})$ where,

1. N, Φ : as before.
2. $\mathbb{T}$: sets of families of latency functions T.

Pre-Orchnets

Call **pre-OrchNet** a tuple $\mathbb{N} = (N, \Phi, \mathbb{T}, \mathbb{T}_{\mathrm{init}})$ where,

1. N, Φ : as before.
2. $\mathbb{T}$: sets of families of latency functions T.
3. $\mathbb{T}_{\mathrm{init}}$: sets of families of initial date functions T_{init}.

Pre-Orchnets

Call **pre-OrchNet** a tuple $\mathbb{N} = (N, \Phi, \mathbb{T}, \mathbb{T}_{\text{init}})$ where,

1. N, Φ : as before.
2. $\mathbb{T}$: sets of families of latency functions T.
3. $\mathbb{T}_{\text{init}}$: sets of families of initial date functions T_{init}.

Pre-Orchnets

Call **pre-OrchNet** a tuple $\mathbb{N} = (N, \Phi, \mathbb{T}, \mathbb{T}_{\text{init}})$ where,

1. N, Φ : as before.
2. $\mathbb{T}$: sets of families of latency functions T.
3. $\mathbb{T}_{\text{init}}$: sets of families of initial date functions T_{init}.

Write $\mathcal{N} \in \mathbb{N}$ if there exists $T \in \mathbb{T}$ and $T_{\text{init}} \in \mathbb{T}_{\text{init}}$ s.t.

$$\mathcal{N} = (N, \Phi, T, T_{\text{init}})$$

Pre-Orchnets: Order Relation

For two families T and T' of latency functions, write

$$T \geq T'$$

if $\forall \omega \in \Omega, \forall t \in \mathcal{T} \implies \tau_t^\omega \geq \tau_t'^\omega$.

Pre-Orchnets: Order Relation

For two families T and T' of latency functions, write

$$T \geq T'$$

if $\forall \omega \in \Omega, \forall t \in \mathcal{T} \implies \tau_t^\omega \geq \tau_t'^\omega$.

For $\mathcal{N}, \mathcal{N}' \in \mathbb{N}$, write

$$\mathcal{N} \geq \mathcal{N}'$$

if $T \geq T'$ and $T_{\mathrm{init}} \geq T'_{\mathrm{init}}$ both hold.

Monotony: Definition

Monotony: Definition

Pre-Orchnet $\mathbb{N} = (N, \Phi, \mathbb{T}, \mathbb{T}_{\text{init}})$ is called **monotonic** if, $\forall \mathcal{N}, \mathcal{N}' \in \mathbb{N}$ *s.t.* $\mathcal{N} \geq \mathcal{N}'$,

$$E_\omega(\mathcal{N}) \geq E_\omega(\mathcal{N}')$$

holds.

A Sufficient Condition for Monotony..

A Sufficient Condition for Monotony..

Pre-Orchnet $\mathbb{N} = (N, \Phi, \mathbb{T}, \mathbb{T}_{\text{init}})$ is monotonic if:

$$\forall \mathcal{N} \in \mathbb{N}, \quad \forall \omega \in \Omega, \ \forall \overline{\kappa} \in \overline{\mathcal{V}}(N),$$

$$E_\omega(\overline{\kappa}, \mathcal{N}) \geq E_\omega(\overline{\kappa}(\mathcal{N}, \omega), \mathcal{N})$$

where $\overline{\mathcal{V}}(N)$ is the set of all maximal configurations of N.

A Sufficient Condition for Monotony..

Pre-Orchnet $\mathbb{N} = (N, \Phi, \mathbb{T}, \mathbb{T}_{\text{init}})$ is monotonic if:

$$\forall \mathcal{N} \in \mathbb{N}, \quad \forall \omega \in \Omega, \ \forall \overline{\kappa} \in \overline{\mathcal{V}}(N),$$

$$E_\omega(\overline{\kappa}, \mathcal{N}) \geq E_\omega(\overline{\kappa}(\mathcal{N}, \omega), \mathcal{N})$$

where $\overline{\mathcal{V}}(N)$ is the set of all maximal configurations of N.

Proof: Let $\mathcal{N}' \geq \mathcal{N}$, then

A Sufficient Condition for Monotony..

Pre-Orchnet $\mathbb{N} = (N, \Phi, \mathbb{T}, \mathbb{T}_{\text{init}})$ is monotonic if:

$$\forall \mathcal{N} \in \mathbb{N}, \quad \forall \omega \in \Omega, \; \forall \overline{\kappa} \in \overline{\mathcal{V}}(N),$$

$$E_\omega(\overline{\kappa}, \mathcal{N}) \geq E_\omega(\overline{\kappa}(\mathcal{N}, \omega), \mathcal{N})$$

where $\overline{\mathcal{V}}(N)$ is the set of all maximal configurations of N.

Proof: Let $\mathcal{N}' \geq \mathcal{N}$, then

$$E_\omega(\mathcal{N}') = E_\omega(\overline{\kappa}(\mathcal{N}', \omega), \mathcal{N}')$$

A Sufficient Condition for Monotony..

Pre-Orchnet $\mathbb{N} = (N, \Phi, \mathbb{T}, \mathbb{T}_{\text{init}})$ is monotonic if:

$$\forall \mathcal{N} \in \mathbb{N}, \quad \forall \omega \in \Omega, \ \forall \overline{\kappa} \in \overline{\mathcal{V}}\,(N)\,,$$

$$E_\omega(\overline{\kappa}, \mathcal{N}) \geq E_\omega(\overline{\kappa}(\mathcal{N}, \omega), \mathcal{N})$$

where $\overline{\mathcal{V}}\,(N)$ is the set of all maximal configurations of N.

Proof: Let $\mathcal{N}' \geq \mathcal{N}$, then

$$E_\omega(\mathcal{N}') = E_\omega(\overline{\kappa}(\mathcal{N}', \omega), \mathcal{N}') \quad \geq \quad E_\omega(\overline{\kappa}(\mathcal{N}', \omega), \mathcal{N})$$

A Sufficient Condition for Monotony..

Pre-Orchnet $\mathbb{N} = (N, \Phi, \mathbb{T}, \mathbb{T}_{\text{init}})$ is monotonic if:

$$\forall \mathcal{N} \in \mathbb{N}, \quad \forall \omega \in \Omega, \ \forall \overline{\kappa} \in \overline{\mathcal{V}}(N),$$

$$E_\omega(\overline{\kappa}, \mathcal{N}) \geq E_\omega(\overline{\kappa}(\mathcal{N}, \omega), \mathcal{N})$$

where $\overline{\mathcal{V}}(N)$ is the set of all maximal configurations of N.

Proof: Let $\mathcal{N}' \geq \mathcal{N}$, then

$$\begin{aligned} E_\omega(\mathcal{N}') = E_\omega(\overline{\kappa}(\mathcal{N}', \omega), \mathcal{N}') &\geq E_\omega(\overline{\kappa}(\mathcal{N}', \omega), \mathcal{N}) \\ &\geq E_\omega(\overline{\kappa}(\mathcal{N}, \omega), \mathcal{N}) = E_\omega(\mathcal{N}) \end{aligned}$$

A Necessary Condition ..

A Necessary Condition ..

If the sufficient condition

$$\forall \mathcal{N} \in \mathbb{N}, \quad \forall \omega \in \Omega, \forall \overline{\kappa} \in \overline{\mathcal{V}}\left(N\right),$$

$$E_\omega(\overline{\kappa}, \mathcal{N}) \geq E_\omega(\overline{\kappa}(\mathcal{N}, \omega), \mathcal{N})$$

is violated,

A Necessary Condition ..

If the sufficient condition

$$\forall \mathcal{N} \in \mathbb{N}, \quad \forall \omega \in \Omega, \forall \overline{\kappa} \in \overline{\mathcal{V}}(N),$$

$$E_\omega(\overline{\kappa}, \mathcal{N}) \geq E_\omega(\overline{\kappa}(\mathcal{N}, \omega), \mathcal{N})$$

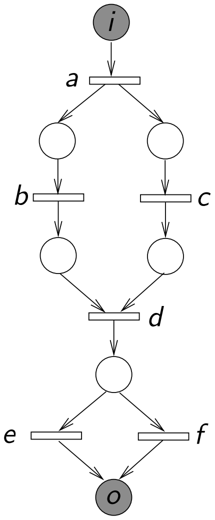
is violated, and for any two OrchNets $\mathcal{N}, \mathcal{N}'$ s.t $\mathcal{N} \in \mathbb{N}$,

$$\mathcal{N}' \geq \mathcal{N} \implies \mathcal{N}' \in \mathbb{N}$$

holds, then $\mathbb{N}$ is not monotonic.

A structural condition for monotony...

Workflow nets (WFnets)



We consider safe WF nets, without any loops.

We consider safe WF nets, without any loops.

$(W,$

We consider safe WF nets, without any loops.

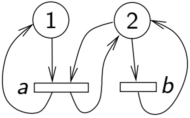
$$(W, \Phi, \mathbb{T}, \mathbb{T}_{\mathrm{init}}) : \quad \textit{pre-WFnet}$$

We consider safe WF nets, without any loops.

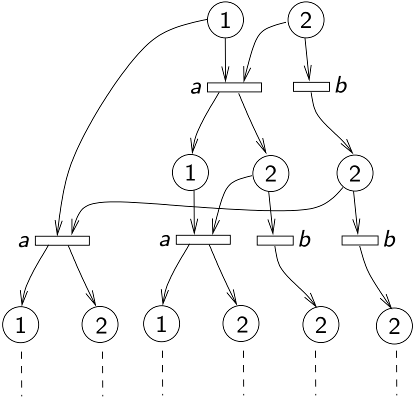
$$(W, \Phi, \mathbb{T}, \mathbb{T}_{\text{init}}) : \quad \textit{pre-WFnet}$$

Unfolding W gives the occurrence net N_W and a corresponding Orchnet

$$(N_W, \Phi_W, \mathbb{T}_W, \mathbb{T}_{\text{init}})$$



N



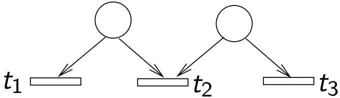
U_N

Clusters.

For a safe net N, a *cluster* is a minimal set $\mathbf{c}$ of places and transitions of N such that

$$\forall t \in \mathbf{c} \implies {}^{\bullet}t \subseteq \mathbf{c} \quad , \quad \forall p \in \mathbf{c} \implies p^{\bullet} \subseteq \mathbf{c}$$

Clusters



t_1, t_2, t_3 are in the same cluster

Sufficient Condition for Monotony of WFnets

W: WFnet, N_W: unfolding of W.

Sufficient Condition for Monotony of WFnets

W: WFnet, N_W: unfolding of W.

Pre-Orchnet $\mathbb{N}_W = (N_W, \Phi_W, \mathbb{T}_W, \mathbb{T}_{\text{init}})$ is monotonic if every cluster c of W satisfies:

Sufficient Condition for Monotony of WFnets

W: WFnet, N_W: unfolding of W.

Pre-Orchnet $\mathbb{N}_W = (N_W, \Phi_W, \mathbb{T}_W, \mathbb{T}_{\text{init}})$ is monotonic if every cluster c of W satisfies:

$$\forall t_1, t_2 \in c,\ t_1 \neq t_2 \implies {t_1}^\bullet = {t_2}^\bullet$$

Only a very restricted class of nets are indeed monotonic.

Conditional Monotony..

Conditional Monotony: Compare execution times only for identical responses.

In Conclusion..

- Identified and defined the problem of monotony in service compositions.
- Insights and reconsideration into the formulation of contracts.

Future Work..

- Extend the notion of monotony to probabilistic contracts.

Future Work..

- Extend the notion of monotony to probabilistic contracts.
- Consider more, QoS parameters in our study.

Thank you..