Law and Partial Order
Nonsequential Behaviour and Probability in Asynchronous Systems

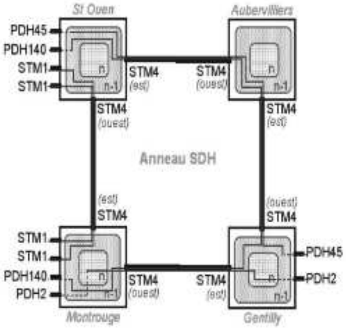
Stefan Haar

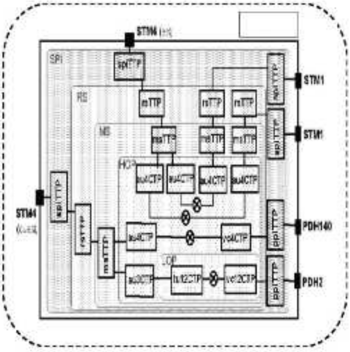
Oct 30, 2008

Contents

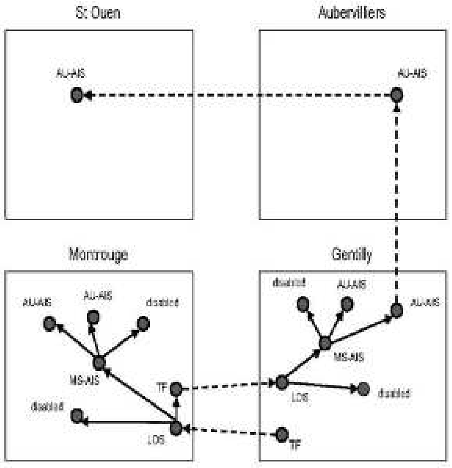
Contents

Fault Diagnosis for Networks *(MAGDA)*

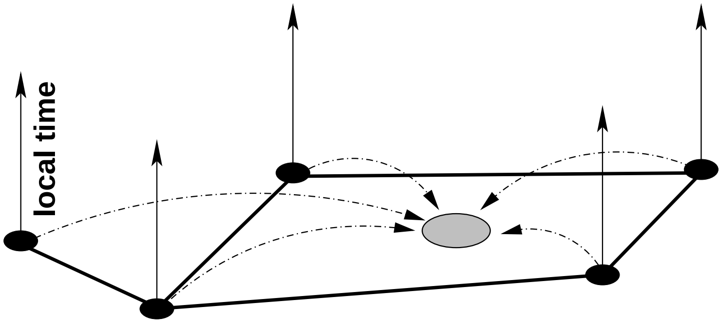




Fault Diagnosis for Networks *(MAGDA)*

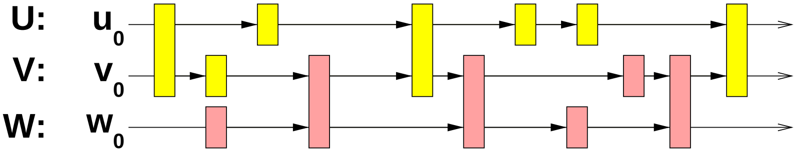


Fault Diagnosis for Networks



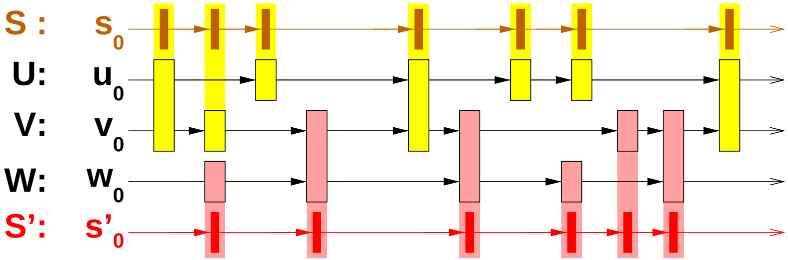
Centralized Diagnoser observes asynchronous alarm streams

Traces



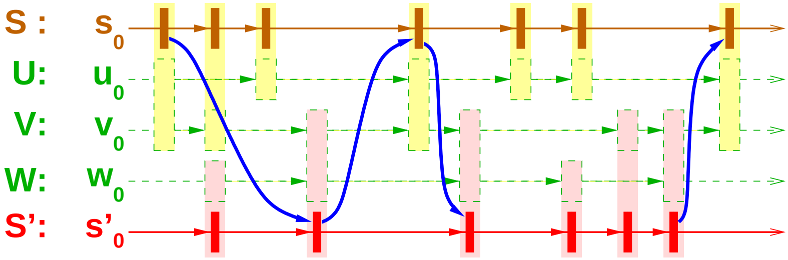
traces ≡ partial orders of tiles

Diagnosis via Synchronization



Correlate traces and observations

Diagnosis via Synchronization

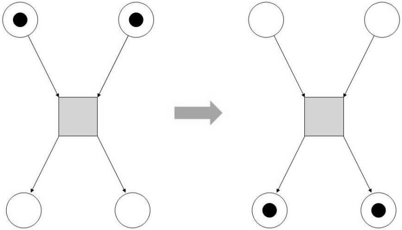


Project to traces, add precedence ordering in alarm patterns

Data structure ??

Share joint prefixes $\longrightarrow$ use Occurrence Nets

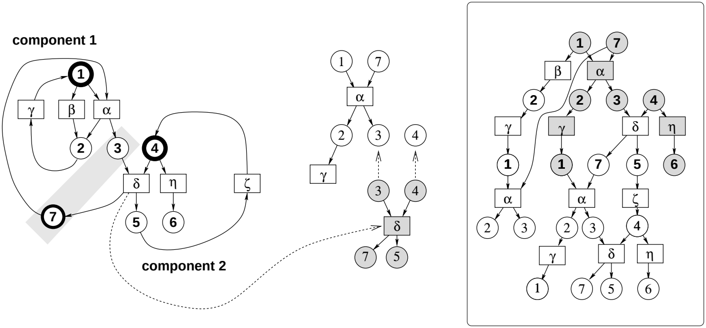
Asynchronous Model: Petri nets



Ingredients

- Transition set $\mathcal{T}$
- Place set $\mathcal{P}$, $\mathcal{P} \cap \mathcal{T} = \emptyset$
- Flow relation $F \subseteq (\mathcal{P} \times \mathcal{T} \cup \mathcal{T} \times \mathcal{P})$
- Marking M: Multiset over / Subset of $\mathcal{P}$

PN Unfoldings and Diagnosis (BFHJ 2003)

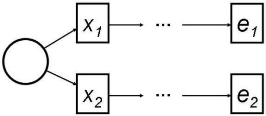


Unfoldings: from PNs to ONs

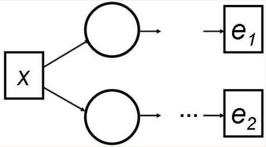
Relations

- **Order**, $e_1 < e_2$:

- **Conflict**, $e_1 \# e_2$:



- **Concurrency**, $e_1 \# e_2$:



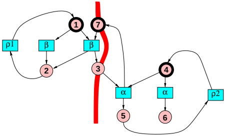
Unfoldings: from PNs to ONs

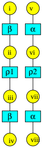
$ON = (\mathcal{B}, \mathcal{E}, G, \mathbf{c}^*)$ is an occurrence net iff:

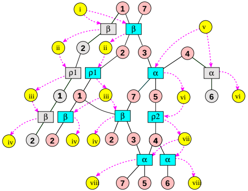
1. No self-conflict: $\forall x \in \mathcal{B} \cup \mathcal{E} : \neg[x \# x]$;
2. $\leqslant$ is a partial order: $\forall x \in \mathcal{B} \cup \mathcal{E} : \neg[x < x]$;
3. finite histories: $\forall x \in \mathcal{B} \cup \mathcal{E} : |\{x' \mid x' < x\}| < \infty$;
4. no backward branching: $\forall b \in \mathcal{B} : |{}^\bullet b| \leq 1$.
5. initial cut: $\mathbf{c}^* := \min(ON) \subseteq \mathcal{B}$.

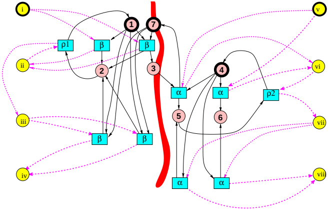
- **Configuration:** conflict free, downward closed set $\mathbf{c}^* \subseteq \kappa \subseteq \mathcal{B} \cup \mathcal{E}$;
- **Run:** $\subseteq$-maximal configuration ω

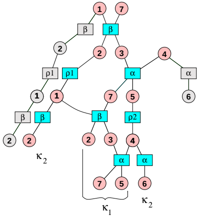
Unfoldings and Diagnosis











Unfoldings and Diagnosis

- Wanted: Set $\mathbf{diag}(\mathcal{A})$ of all configurations that explain alarm pattern $\mathcal{A}$
- Solution: using product net $\mathcal{N} \times \mathcal{A}$,
  $\mathbf{C} \in \mathbf{diag}(\mathcal{A})$ iff

$$\exists\ \overline{\mathbf{C}} \in \mathbf{config}(\mathcal{U}_{\mathcal{N}\times\mathcal{A}}) : \mathbf{proj}_{\mathcal{N}}(\overline{\mathbf{C}}) = \mathbf{C},\ \mathbf{proj}_{\mathcal{A}}(\overline{\mathbf{C}}) = \mathcal{A}.$$

- Online pruning
- Diagnosability ?

Contents

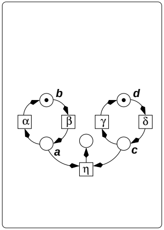
What is required of the system for diagnosis to work ?

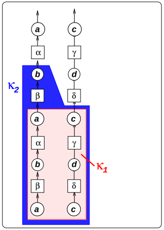
- Need **observability** : no *invisible* cycles
- Need **diagnosability**: Unobservable e is **not diagnosable** in live language $\mathcal{L}$ iff $\exists\ w_N, w_Y \ \in \ \mathcal{L}$:
  - w_Y contains e and w_N does not;
  - w_Y arbitrarily long after e;
  - $P_O(w_Y) = P_O(w_N)$.

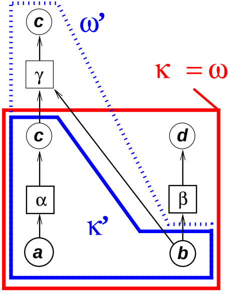
  where $P : \mathfrak{A}^* \rightarrow O^*$ is the projection to observable words.

Weak vs. Strong observability / diagnosability

Analogous to the FSM case, but languages to be chosen with care:







Often, both hold only for **progressive** configurations

Weak vs. Strong observability / diagnosability

$\mathbf{K}$ is progressive iff its upper and lower heights are equal:

$$\sup\{n \in \mathbb{N} \mid \exists \omega \in \Omega : \ \mathbf{K} \cap \rho_n = \omega \cap \rho_n\} = \|\mathbf{K}\|,$$

where $\|\mathbf{K}\|$ is the length of $\mathbf{K}$'s longest causal chain.

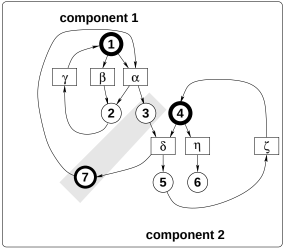
- $\mathcal{N}$ is observable/diagnosable iff the language of its configurations is, and
- **weakly** observable/diagnosable iff the language of its **progressive** configurations is.

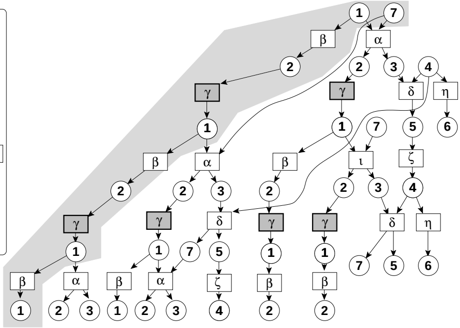
Characterization via checking of cycles.

Verifying Diagnosability

- Strong observability/diagnosability
  - Sufficient **and** necessary conditions: costly
  - We do not take advantage of partial orders
  - Sufficient condition: if there is no invisible/indeterminate T-invariant, $\mathcal{N}$ is strongly observable/diagnosable
- Weak observability/diagnosability
  - Can explore unfolding structure and exploit **covering** relation

Verifying Weak observability/diagnosability

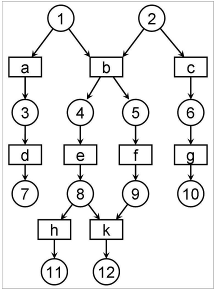




Verifying Weak observability/diagnosability

- ... leads to reasoning of the type
  'if x occurs, then y must have occured/is occurring/will occur'
- Can we make this precise ? YES !
- Can we compute it ? YES !

Covering relation



$$\forall\, \omega : \quad k \in \omega \Rightarrow e \in \omega \Rightarrow b \in \omega$$

$$a \in \omega \iff \neg(b \in \omega) \iff c \in \omega$$

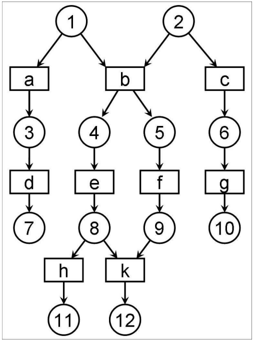
$$\forall\, \omega : e \in \omega \iff f \in \omega$$

Covering relation

In ON :

- x **implies** or **covers** y, written $x \triangleright y$, iff $z \# y \Rightarrow z \# x$.
- **THM:** $x \triangleright y$ holds iff for all runs ω $x \in \omega \Rightarrow y \in \omega$.
- $y < x$ implies $x \triangleright y$
- $y \triangleright x$ compatible also with $y < x$ and y **co** x
- $\triangleright[x]$ is a configuration.

Covering relation



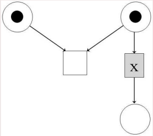
$$\triangleright[h] = \{b, e, f, h\} \quad , \quad \triangleright[k] = \{b, e, f, k\}$$
$$\triangleright[a] = \triangleright[d] = \triangleright[c] = \triangleright[g] \quad = \quad \{a, d, c, g\}$$

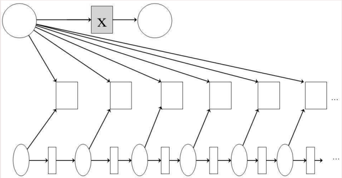
Covering relation

- $\triangleright$, it suffices to inspect immediate conflicts:

$$\#[x] \;=\; \{z' \mid \exists y \in \#_\mu[x] : y \leqslant z\}.$$

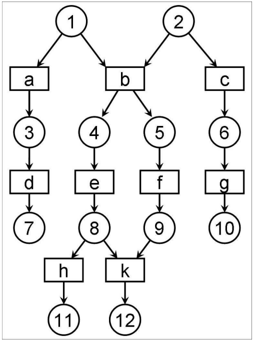
- Caveat: $\#_\mu[x]$ is not necessarily finite ...





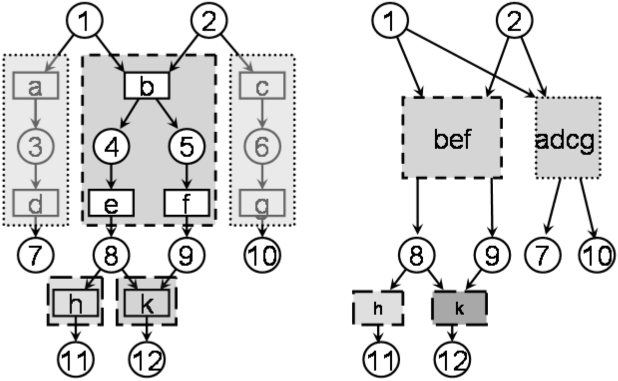
... but $\triangleright$ can be computed on a bounded prefix

Facets: When $\triangleright$ holds both ways



$$\triangleright[b] = \triangleright[e] = \triangleright[f] = \{b, e, f\}$$

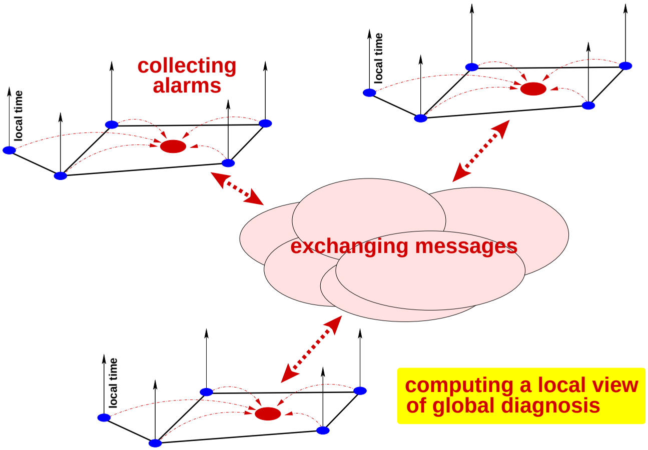
Facets



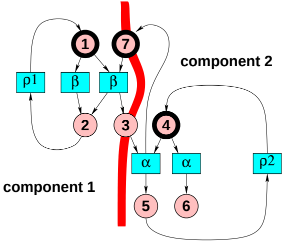
Covering

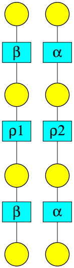
- Covering relation effectively computable
- $\triangleright$ formalizes occurrence dependencies under progress
- Helps search for minimal observability (i.e. which events must be visible to allow detection) for a given task, such as control, diagnosis, verification, (test ?), ...
- Large unfoldings can be reduced by facet abstraction
- To o :
  - Read nets
  - Probability
  - Link with temporal logics

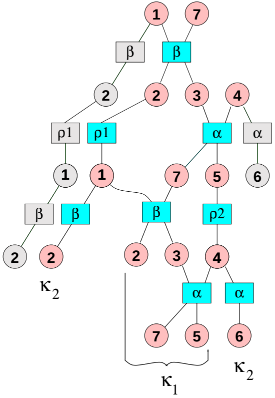
To fight prefix explosion ...



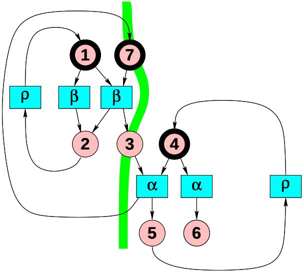
... distribute the unfolding !

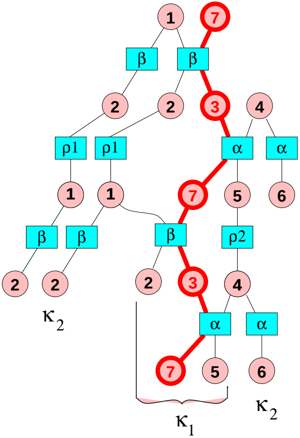






Distributed Computation of Explanations





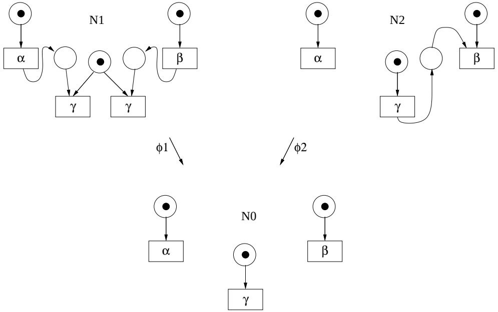
interaction

Distributed Unfolding

Formalizing

- Want: Decomposition of Petri nets (GGs, ... ?) that allows to unfold local views + to avoid computation of global unfolding
- Fabre/Benveniste/Haar/Jard CONCUR 03 + JDEDS 2005
- *Koenig/Baldan/Haar FOSSACS 06*: Use Pullbacks and limit preservation under coreflections
- Need to use interleavings rather than event structures on interfaces

Projections



- … fail for ON's and Event Structures !
- Conclusion: use interleavings rather than event structures communication between local supervisors and adjustment of local views

References

- Benveniste, Fabre, Haar, Jard :
  - Asynchronous diagnosis for PN TAC 2003
  - Probability TAC 2003 etc
  - Distributed diagnosis JDEDS 2005
- Distributed unfolding of PNs:
  Baldan, Haar, König FOSSACS 2006
- Asynchronous diagnosis for Graph grammars: Baldan, Chatain, Haar, König CONCUR 2008
- Haar: Diagnosability, Covering CDC 2003 + 2007, submitted

Contents

Why Asynchronous Stochastic Processes ?

- **Diagnosis** may result in ambiguity: the same alarm pattern may be explained by several different runs of the system
- In that case, choose the **most likely one**
- Therefore, need a model for stochastic processes that reflects asynchronous dynamics
- Classical stochastic processes have one-dimensional trajectories; **need:** model with partially ordered realisations
- Here: no *external* time parameter, need to find *internal* process time

Markov Nets (BHF 2003, Abbes et al.)

- Desirable: analogon of (discrete time) Markov chains : $(X_n)_{n\in\mathbf{N}}$ such that $\mathcal{L}(X_{n+k} \mid X_1, \ldots, X_n) = \mathcal{L}(X_{n+k} \mid X_n)$
- Allows e.g. characterization of stationary laws & asymptotic behaviour
- Markov processes reflect 'absence of memory' of the system ...
- ... and allow for recursive computation of probabilities , with compact representation of laws

Stopping Times

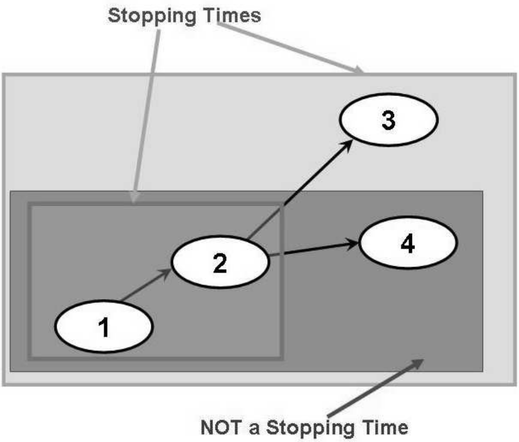
- Random time index $\tau : \Omega \to \mathbf{T}$ depending on $(X_t)_{t \in \mathbf{T}}$ s.th.

$$\forall\ t \ \in \ \mathbf{T}: \ \{\omega \in \Omega \mid \tau(\omega) \ \leqslant \ t\} \ \in \ \mathcal{F}_t$$

- Strong MP , for $\mathbf{T} = \mathbb{N}$: if τ is a stopping time, then $\forall, A: \ \mathbf{P}(X_{n+1} \in A \mid \ X_1, \ldots, X_n) = \mathbf{P}(X_{n+k} \in A \mid \ X_n)$
- Examples of stopping times:
  - The fire engine must be maintenanced and tested *on November 7*
  - The fire engine must be maintenanced and tested *the day following its use*
- NOT a stopping time:
  - The fire engine must be maintenanced and tested *the day before a fire breaks out*

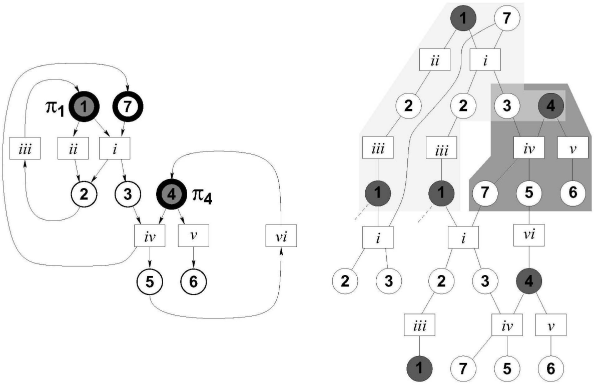
Stopping Times

In DTMC, stopping times are *closed under conflict*



Stopping Times

Conflict-closure for stopping times in PN !



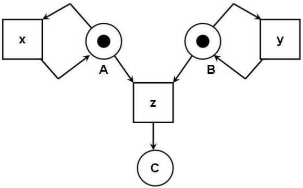
Markov Nets (BHF 2003, Abbes et al.)

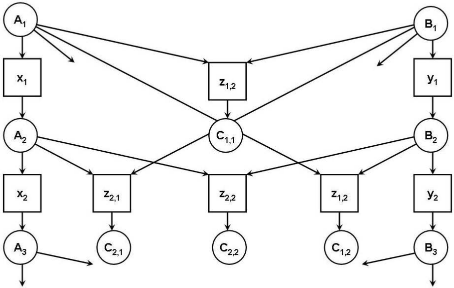
- Probabilistic routing on each place
- Renormalization to obtain the prob. of occurrence for each event inside the first stopping time τ_1 = *first layer*
- Redo for next layer (better (S. Abbes): next *branching cell*)
- *Strong Markov Property:*

$$\mathbf{P}\left(B \mid \mathcal{F}_\tau\right) = \mathbf{P}\left(B \mid \mathcal{X}_\tau\right).$$

Stopping Times

... may not be finite, so probabilization *may fail*:

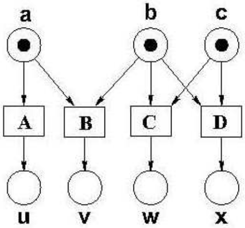




Cluster Semantics

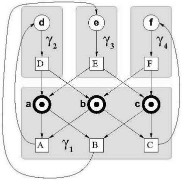
Leaving the Markov net approach, more is possible:
Consider Conflict cluster or t:
smallest subnet $\gamma = \gamma(t) \subseteq \mathcal{T} \cup \mathcal{P}$ such that

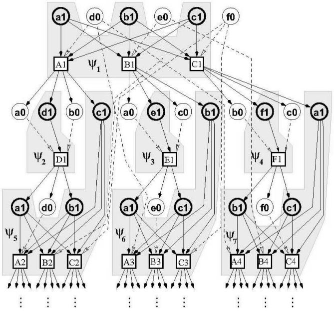
1. $t \in \gamma$;
2. if $t' \in \mathcal{T} \cap \gamma$, then ${}^\bullet t' \subseteq \gamma$; and
3. if $p \in \mathcal{P} \cap \gamma$, then $p^\bullet \subseteq \gamma$.



Cluster Semantics

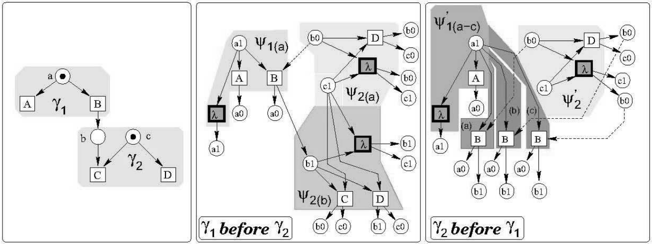
... new unfolding semantics, still yielding ONs





Cluster Semantics

Time strikes back: need scheduling policies



Cluster Semantics

- Probabilistic unfolding for all net structures
- Not even safeness required
- Natural *counting time*: number of policy rounds
- Stopping times are exactly the tile-respecting prefixes
- Markovian in time ...
- ... and space: choice of transition set can follow a Markov field $\rightarrow$ *right* conditional independence of transitions
- Details in FI 2002/2003

Contents

Time and probability

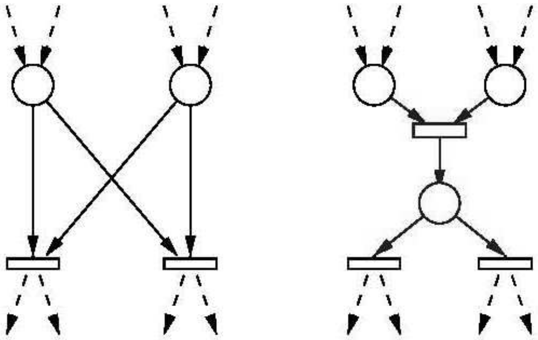
- In timed Petri nets, logical choice is intertwined with durations
- Random durations probabilize behaviour
- Standard approach: *stochastic Petri nets*
  - (essentially) all durations memoryless $\longrightarrow$ exp
  - Yields CTMC, but excludes many laws (gamma, heavy tail, ...)
- Idea (Gaujal/Haar/Mairesse 2003):
  - Allow arbitrary $(0, \infty)$ duration laws
  - Establish *renewal* markings, and ...
  - ... thus obtain asymptotic throughput results
- For this, need structural restrictions

Time and probability

BLOC Block any (non-branching) transition b of live and bounded net $\mathcal{N}$; then $\mathcal{N}$ eventually stops in unique marking M_b

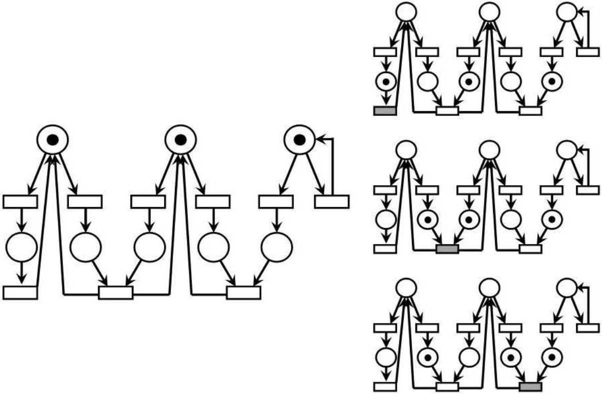
- Free choice nets satisfy BLOC ...
- ... and hardly any other class does !
- Under *reasonable* assumptions, $\mathcal{N}$ goes into a blocked marking a.s.
- M_b's are renewal points $\longrightarrow$ win !!

Free Choice Nets



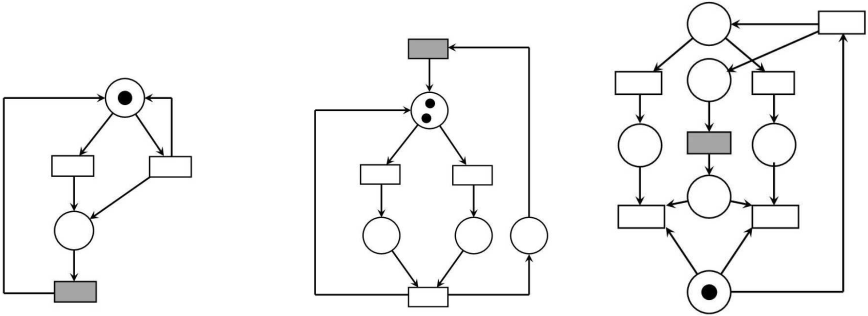
Free Choice Nets + Blocking

Transitions and associated blocked markings



Free Choice Nets + Blocking

Boundedness, liveness and FC **cannot** be dropped



Testing Partial Order Input/Output Automata

Given:

- A formal **specification** $\mathcal{S}$: black box, only I/O
- **Implementation** $\mathcal{I}$ of $\mathcal{S}$, which may or may not be correct

Want:

automatically generate test sequences to prove **conformance** of the implementation to the specification

Testing: Concurrent multi-port machines

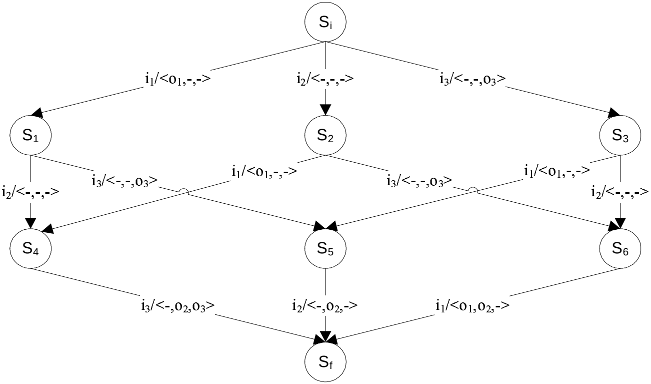
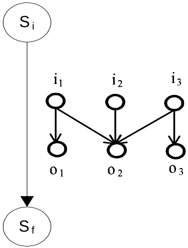


Figure: (Partial) Multiports Deterministic FSM.

Testing bPOIOA

The "classical" approach to testing automata does not work with bPOIOA because the causal relationships are not observed.



Solution

- Delay input on one port
- Observe outputs
- Send last input
- Observe outputs
- repeat

Checking Sequence construction

Theorem

Given an bPOIOA of n states and t transitions having an adaptive checking sequence, assuming that the implementation is in the initial state, the following test sequence is a checking sequence of size $O(tpn^3 + pn^4)$

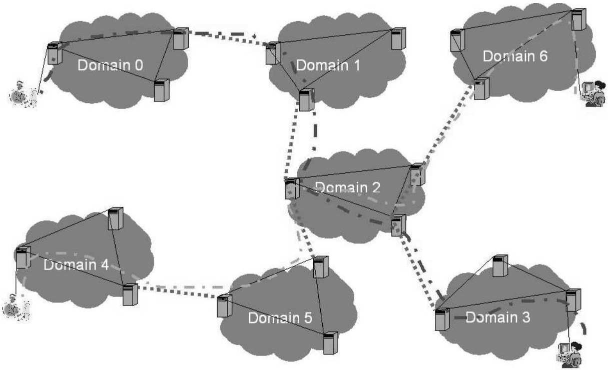
1. *Check all states with the test sequence checking all states*
2. *For all transitions do:*
   1. *transfer to the starting state of the transition*
   2. *check the transition with the test sequence checking transitions*

Test : present and future

- general POs (non-bipartite): 2008
- Weak synchronization at states
- Petri Nets
- References:
  - S. Haar and C. Jard and G.-V. Jourdan: Testing Input/Output Partial Order Automata. *TestCom*, 2007.
  - G. v. Bochmann and S. Haar and C. Jard and G.-V. Jourdan. Testing Systems Specified as Partial Order Input/Output Automata. *TestCom*, 2008.

SWAN I: QoS contracts in heterogeneous Networks

Negotiation + Monitoring



SWAN I: QoS contracts in heterogeneous Networks

- Distributed negotiation of end-to-end QoS for e.g. videoconference
- Monitoring , repair, optimization
- SWAN cooperation with ALU, QosMetrics, FT, LORIA, LIPN/LABRI
- Thesis of H. Pouyllau; ICT, ICWS, Qshine, Annals Telecom

SWAN II: Composition of WS

- Orchestrations (e.g. BPEL ; Orc (UofTexas))
- QoS composite/cumulative
- Challenges:
  - Predict overall latency for contract
  - Find *critical* components
  - Analyze impact (monotonicity etc)
- AES semantics for Orc (Rosario/Benveniste/Haar/Jard WSFM 2007)
- QoS: on-going work with A. Bouillard, S.Rosario, A.Benveniste; ICWS 2007 etc.

Miscellaneous

- AXML: intensional and dynamic data
- Want : analyze and control document flows
- Have: *Datalog* query system for diagnosis (Abiteboul/Abrams/Milo/Haar PODC 2005)
- Cyclic ordering (ROGICS 2008)
- more on PNs

Contents

Trajectory

- PhD Hamburg 1997
- PostDoc Berlin, Nancy, Paris
- INRIA Rennes
- U of Ottawa (sabbatical) 2007
- ALU Bell Labs Ottawa 2008
- now : INRIA Saclay, LSV at ENS Cachan

Projects and other activities

- Participation ALAPEDES, MAGDA, MAGDA2
- SWAN leader
- ASAX
- PhD Thesis Supervision Hélia Pouyllau
- Co-supervisions etc.
- IEEE TAC associate editor
- INRIA GTRI, DRI

Cooperations

- (the above, and ...)
- Stuttgart/Duisburg, Venise/Padova, Pisa
- University of Ottawa
- Queen's University

Present and Future Subjects

- *Partial orders are good for you* !
- Analyze and deduce processes, completeness of model, dynamic topologies
- Tools for reasoning on branching partial orders
- Link between real time and logical evolution
- Probability, Asynchrony and Distribution
- Control and Test
- Civilized Networks and services (contracts, monitoring, ...)

The End

THANKS !