DEPENDABILITY ENGINEERING WITH TIME-DEPENDENT PETRI NETS

(“THE PROBLEM IS CHOICE”)

CONTENTS

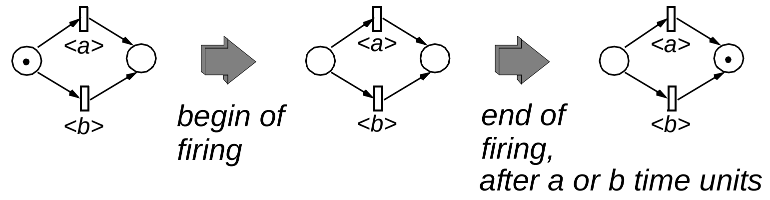
MODEL CLASSES

PETRI NETS

- PLACE/TRANSITION PETRI NET (COLOURED PN)
  - context checking by Petri net theory
  - verification by temporal logics
- TIME-DEPENDENT PN
  - TIME PETRI NET
    - worst-case evaluation
  - STOCHASTIC PETRI NET
    - performance prediction
    - reliability prediction
  - CONTINUOUS PETRI NET
    - ODEs

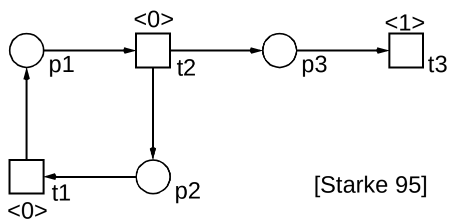
WHICH KIND OF TIME MODEL? (1)

- atomic sequential program parts -> transitions
  - -> *time assigned to transitions*
- as simple as possible
  - -> *timed nets [Ramchandani 74]*
  - -> *duration nets (D nets, DPN)*
- **duration nets**
  - -> *constant times assigned to transitions*
  - -> *token reservation*
  - -> *firing consumes time*



IMMEDIATE TRANSITIONS

- zero (insignificant) time consumption
- **time deadlocks** *(-> **ZENONESS**)*



[Starke 95]

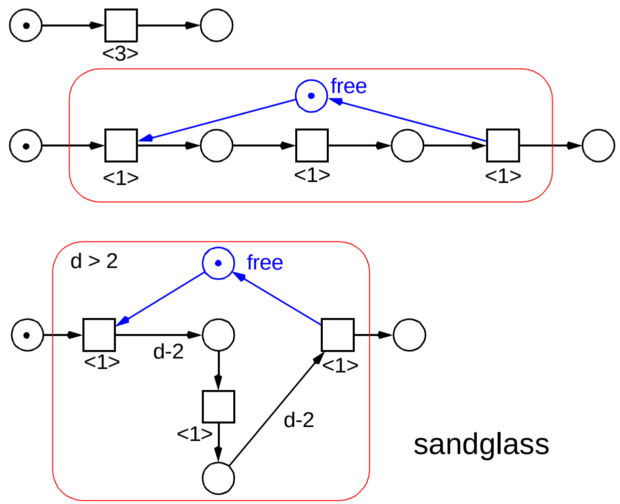
- time deadlock = state from which
  - -> *no transient state is reachable*
  - -> *or: no state is reachable where the system clock is able to advance*
- infinitely many firings in zero times
- inconsistent time constraints !
- How to avoid time deadlocks?
  - -> *invariants ?*
  - -> *OPEN PROBLEM !*

HOW TO ANALYSE DURATION NETS?

- time is running
  - -> *change of the fire rule*

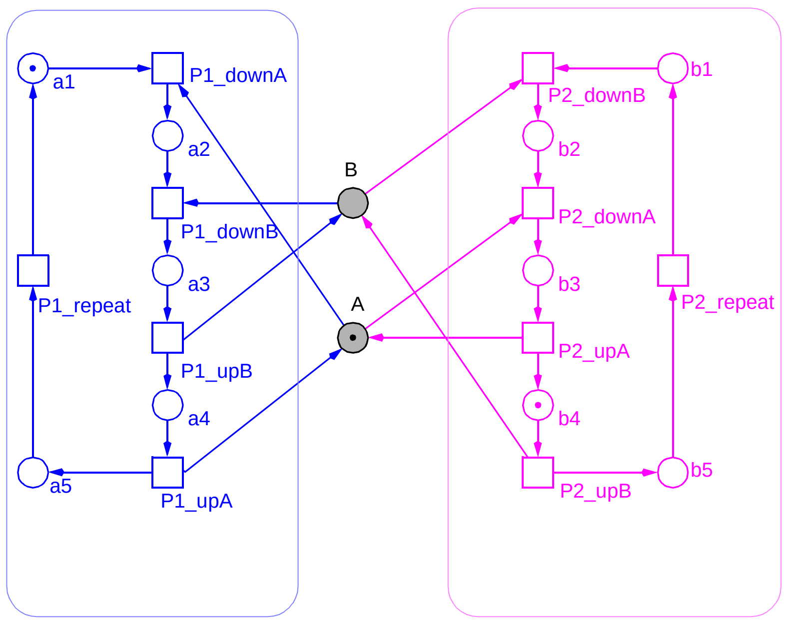
pn		***tpn***
t may fire	->	*t must fire*
single step	->	*maximal step*

- special case:
  duration of all transitions = 1 time unit
  - -> *reachability graph construction under the maximal step firing rule*
- else: transformation into special case



THE INFLUENCE OF TIME
EXAMPLE 1 (SYSTEM DEADLOCK),
PETRI NET

different initial marking !

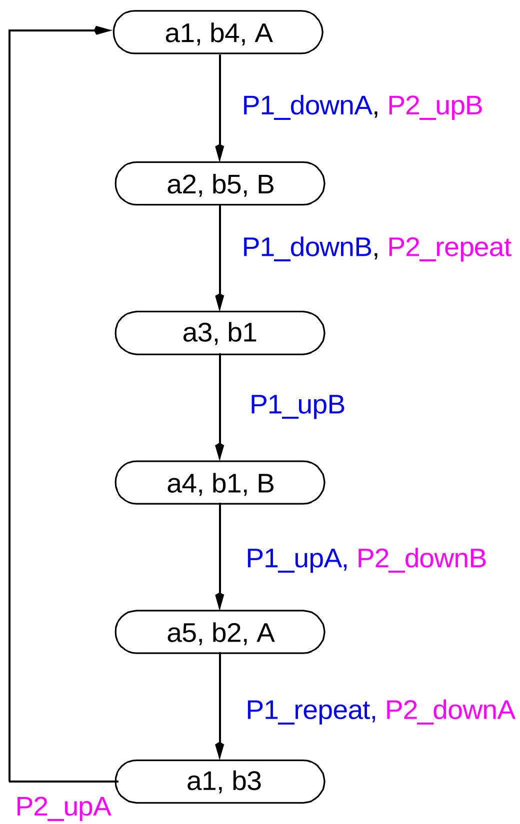


INA

ORD	HOM	NBM	PUR	CSV	SCF	CON	SC	Ft0	tF0	Fp0	pF0	MG	SM	FC	EFC	ES
Y	Y	Y	Y	N	N	Y	Y	N	N	N	N	N	N	N	N	Y

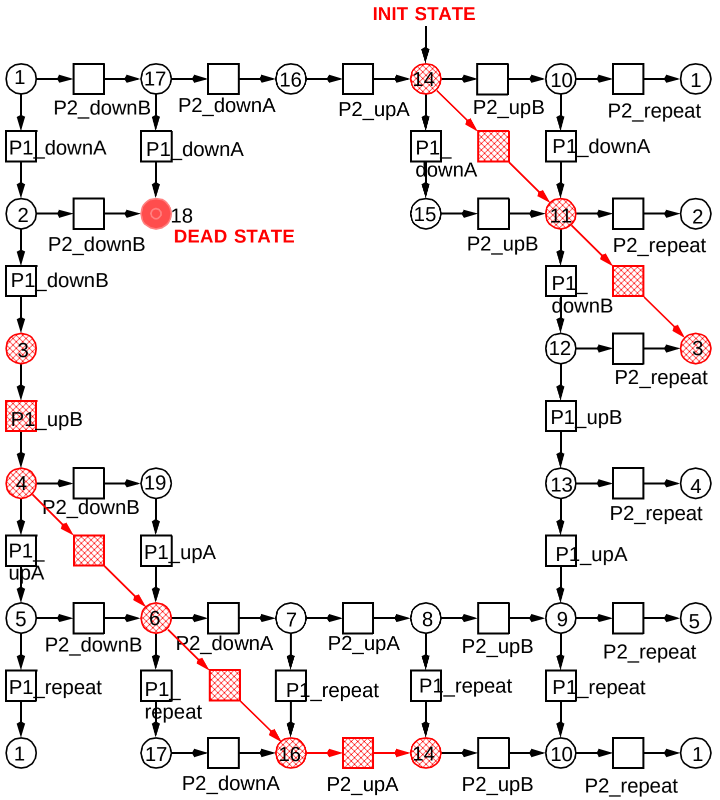
DTP	SMC	SMD	SMA	CPI	CTI	B	SB	REV	DSt	BSt	DTr	DCF	L	LV	L&S
N	Y	Y	N	Y	Y	Y	Y	N	Y	?	N	N	N	N	N

EXAMPLE 1
SYSTEM DEADLOCK,
MAX STEP RG = RG(DPN)



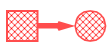
DSt (pn) -> <u>not</u> DSt (tpn)

EXAMPLE 1
SYSTEM DEADLOCK, REACHABILITY GRAPH

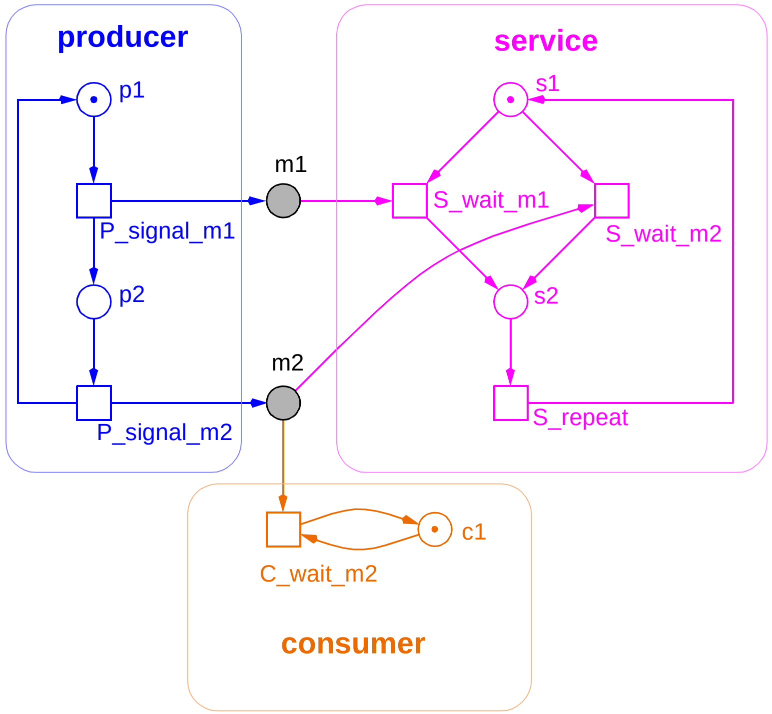


RG (pn)
19 nodes,
32 arcs

RG (tpn)
6 nodes,
6 arcs

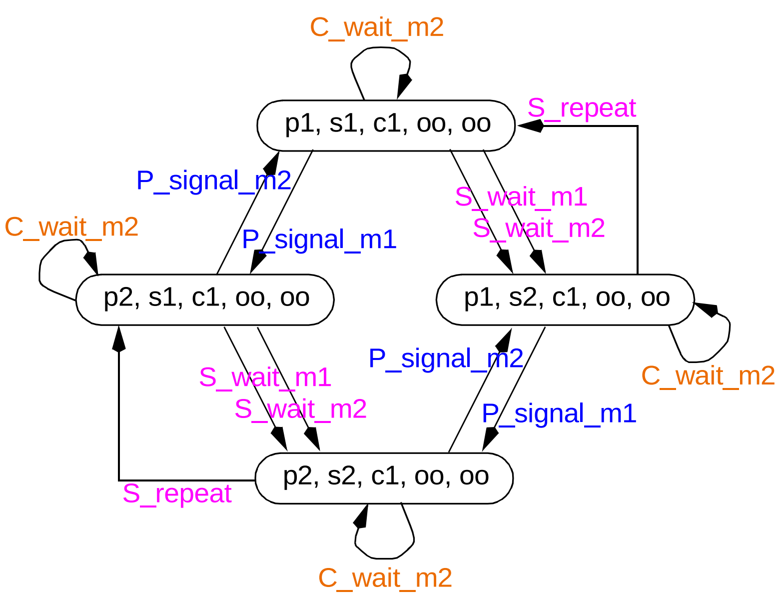


THE INFLUENCE OF TIME, EXAMPLE 2



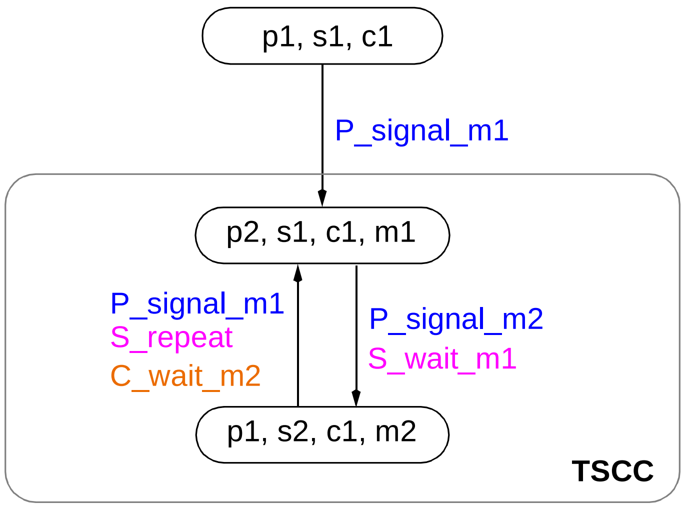
not BND (pn) -> BND (tpn)
not DTr (pn) -> DTr (tpn)

EXAMPLE 2, COVERABILITY GRAPH



- not BND,
  simultaneously unbounded in m1 and m2
- LIVE

EXAMPLE 2, MAX STEP RG = RG(TPN)



- BND,
  - -> *cycle time(p) = 2*
  - -> *cycle time (s) = 2*
  - -> *cycle time (c) = 1*
- not LIVE
  - -> *TSCC does not contain S_wait_m2*
  - -> *S_wait_m2 is m_0-dead*

EXAMPLES, SUMMARY

- example 1
  -> *DSt (pn)* -> *not DSt (tpn)*

- example 2
  -> *not BND (pn)* -> *BND (tpn)*
  -> *not DTr (pn)* -> *DTr (tpn)*

- generally

PN	$\xrightarrow{T \rightarrow \text{TIME}}$	TPN
prop(pn)	$\not\Longleftrightarrow$	prop(tpn)
RG (pn)	$\supseteq$	**RG** (tpn)

- BUT,
  for Petri net based system validation,
  we are only interested in the conclusions

prop(pn) $\xrightarrow{??}$ prop(tpn)

THE INFLUENCE OF TIME ON QUALITATIVE PROPERTIES

TIME-INSENSITIVE RESULTS

- BND (pn) -> BND (tpn) *ok*
- not DSt (pn) -> not DSt (tpn) *ok*
- DTr_{m0} (pn) -> DTr_{m0} (tpn) *ok*

TIME-SENSITIVE RESULTS

- not BND (pn) -> BND (tpn) *ok*
- DSt (pn) -> not DSt (tpn) *ok*
- live (pn) -> not live (tpn) *ko ?*
- REV (pn) -> not REV (TPN) *ko ?*
- not REV (pn) -> REV (tpn) *ok*

SUMMARY

- EF -properties: $\overline{\text{prop}}$ (pn) -> $\overline{\text{prop}}$ (tpn)
- AG EF-properties: prop (pn) <- prop (tpn)

PROBE EFFECT

- **observation** -
  the system exhibits in test mode other
  (less) behaviour than in standard operation mode

- **cause** -
  sw test means (debugger) affect the timing behaviour

- **result** -
  masking of certain types of system behaviour / bugs

  -> *DSt (pn)* -> *not DSt (tpn)*
  -> *live (pn)* -> *not live (tpn)*
  -> *not BND (pn)* -> *BND (tpn)*
  -> *not REV (pn)* -> *REV (tpn)*

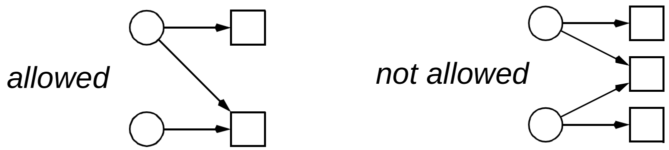
- **consequence** -
  systematic & exhaustive testing
  of concurrent systems is generally impossible

- **wayout** -
  qualitative models considering any timing behaviour

TIME-INVARIANT NET STRUCTURES

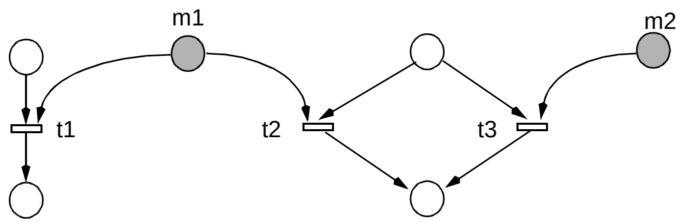
- time-invariant == time independently live

- D nets [Starke 90]
  -> *homogeneous ES nets*



- generalization ?
  -> *behavioural ES nets ?*

- troublemaker - confusing combination of
  channel and control flow conflicts

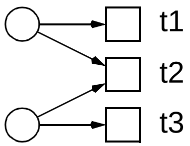


-> “The problem is choice !”

CONFUSION

- concurrency and conflict overlap

  ->



  -> *t1 # t2 and t2 # t3,*
  but t1 concurrent to t3

- **case 1**: t1 < t3

  -> *conflict t2 # t3 disappears,*
  firing of t3 does not involve a conflict decision

- **case 2**: t3 < t1

  -> *conflict t2 # t3 exists,*
  firing of t3 involves a conflict decision

- the interleaving sequences of concurrency
  may encounter different amount of decisions

- an observer outside of the system
  does not know whether a decision took place or not

ARE THERE

TIME-INVARIANT

SOFTWARE STRUCTURES ?

INFLUENCE OF COMMUNICATION PATTERNS ON NET STRUCTURE CLASSES

addressing waiting\	direct / semi-direct-by-sender	indirect / semi-direct-by-receiver
determininistic	EFC	ES
non-deterministic	ES	ICP

- simplified view
  - -> *provided, pre- and postprocesses do not access the same communication object from different control points*

- known to be time-independently live [Starke 90]

  i.e. a live net remains live under any constant delay timing.

INFLUENCE OF COMMUNICATION PATTERNS ON CONFLICT STRUCTURES

<table>
<tr><th>\addressing waiting</th><th>direct / semi-direct-by-sender</th><th>indirect / semi-direct-by-receive</th></tr>
<tr><td>deterministic</td><td rowspan="2">no dynamic channel conflicts</td><td>channel & control flow conflicts appear only separately</td></tr>
<tr><td>non-deterministic</td><td>confusing combination of channel & control flow conflicts possible</td></tr>
</table>

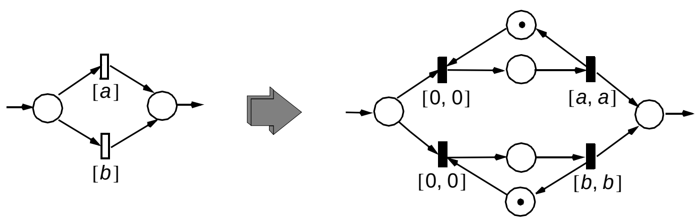
WHICH KIND OF TIME MODEL ? (2)

- ❑ adequate characterization of time consumption
  - -> *alternatives, iterations*
  - -> *time nets, [Merlin 74]*
    interval nets, I nets

- ❑ structural simplicity, e. g. alternative as

duration net
(with token reservation)
(constant times)
(firing consumes time)
working time

interval net
(no token reservation)
(interval times)
(firing itself timeless)
reaction time



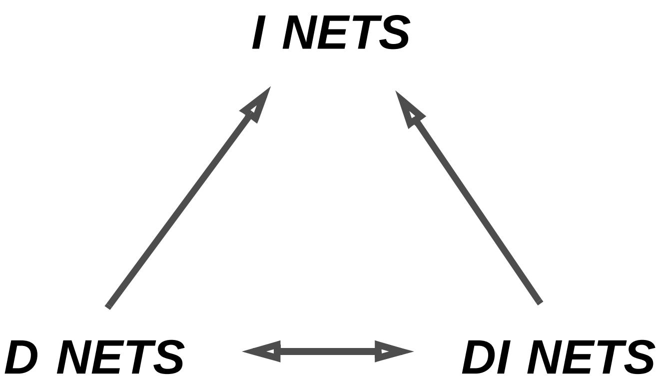
- ❑ **duration interval net**, DI net
  - -> *interval times*
  - -> *with token reservation*
  - -> *firing consumes time*

NON-STOCHASTIC T-TIME-DEPENDENT PETRI NETS, OVERVIEW

firing principle / times	**WORKING TIME** (token reservation)	**REACTION TIME** (no token reservation)
constant	timed nets [Ramchandani 74] ***-> (working time) duration nets*** ***D NETS***	- / - *-> reaction time duration nets* ?
interval	- / - *-> working time interval nets* ***DI NETS***	time nets [Merlin 74] ***-> (reaction time) interval nets*** ***I NETS***

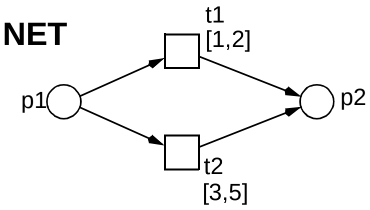
RELATION OF TIME-DEPENDENT PETRI NETS (TRANSITION → TIME)

CONJECTURE !

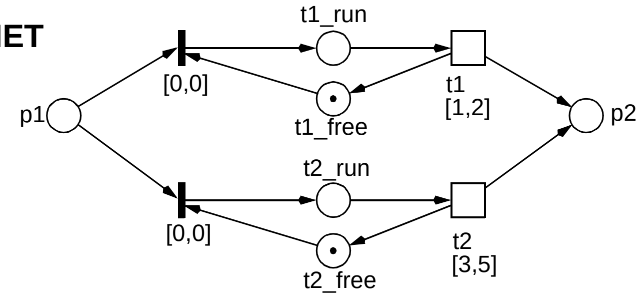


TRANSFORMATION DI NET -> I / D NET

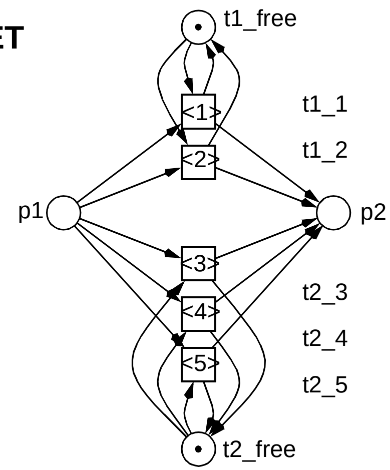
DI NET



I NET

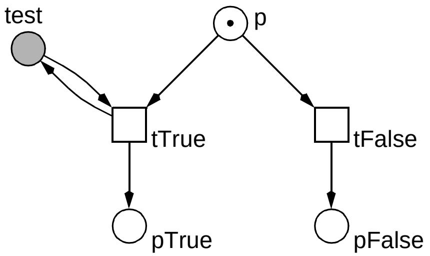


D NET

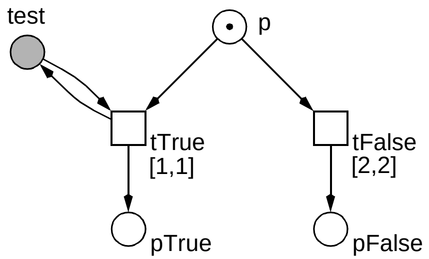


ZERO TEST FOR TURING POWER

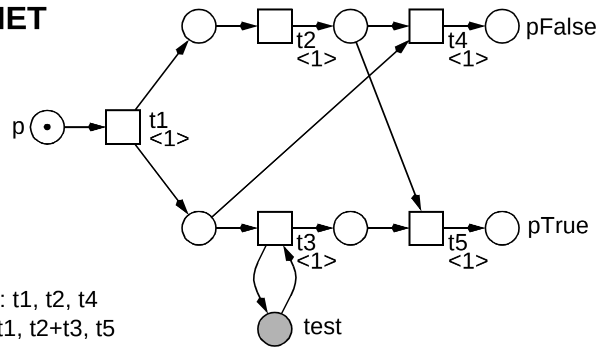
PETRI NET ?



I NET



D NET



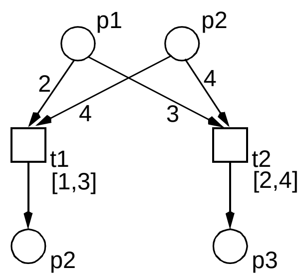
False: t1, t2, t4
True:t1, t2+t3, t5

TIME-INVARIANT NET STRUCTURES (I NETS)

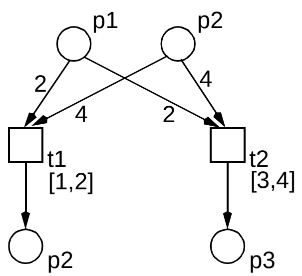
A live Petri net remains live under any timing, [Popova 95]

- if it is persistent,
- if the earliest firing time of all transitions is zero
- if the latest firing time of all transitions is infinite
- if it is an homogeneous & timely homogeneous EFC without purely immediate transitions

not homogeneous



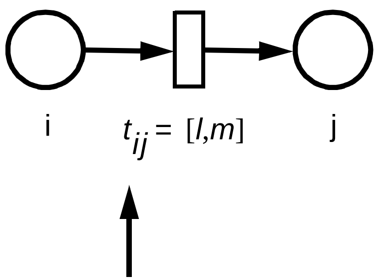
not timely homogeneous



- if it is an homogeneous & timely homogeneous behaviourally free choice net without purely immediate transitions.

  pre(t1) $\cap$ pre(t2) -> AG (enabled(t1) $\Leftrightarrow$ enabled(t2))

WORST-CASE EVALUATION WITH DI NETS, INPUT PARAMETERS

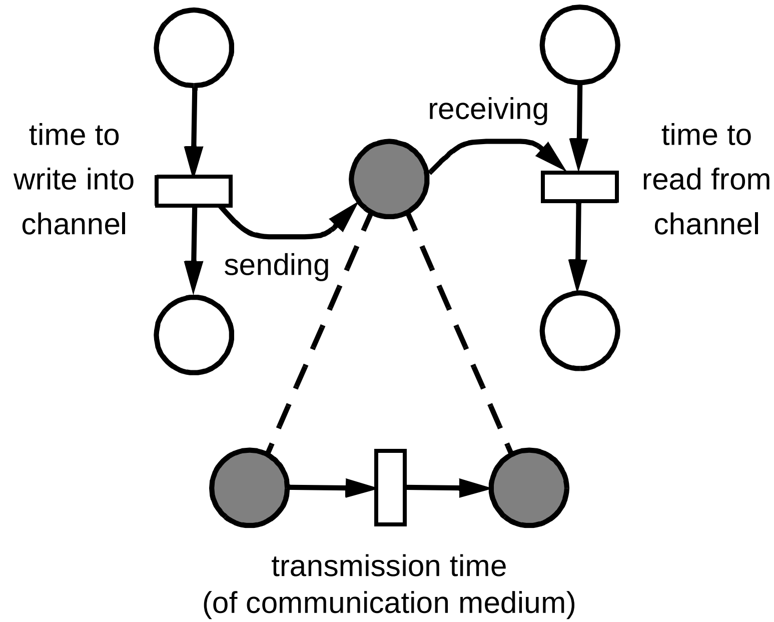


- time consumption of sequential program parts
  - at least ***l*** time units
    - (lower bound of duration time, $low(t_{ij}) = l$)
  - at most ***m*** time units
    - (upper bound of duration time, $upp(t_{ij}) = m$)
  - or any (continuous) time in between

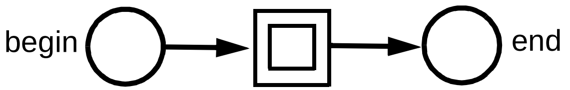
  measured by monitoring OR
  calculated from computer instructions

- no explicit branching probabilities

COMMUNICATION TIME MODEL



WORST-CASE EVALUATION WITH DI NETS OUTPUT PARAMETERS

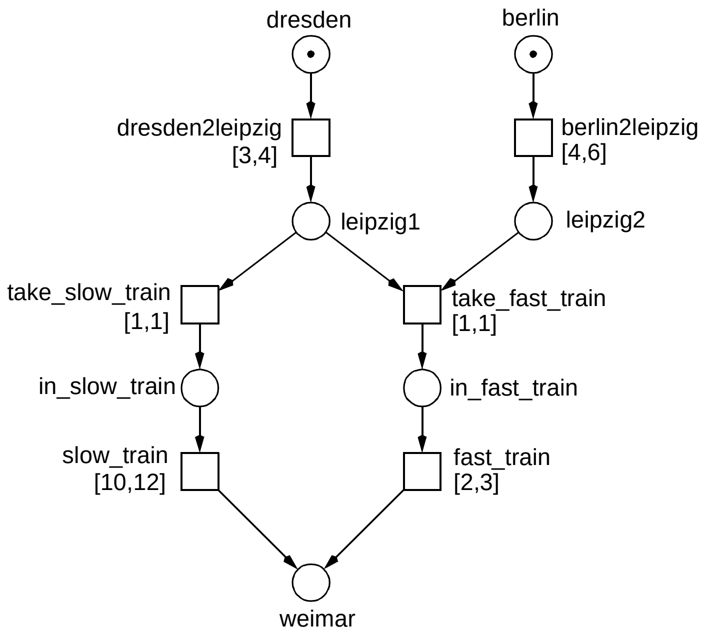


$t_{begin, end} = [min, max]$

- ❑ min execution duration (shortest path), max execution duration (largest path)
- ❑ esp. valuable for systems which require predictable timing behaviour (to meet given deadlines)
- ❑ calculations can be based on discrete reachability graph (only integer states)

  -> *INA*

COMPUTATION OF MINIMAL PATH BY LOWER BOUNDS ONLY, COUNTER EXAMPLE

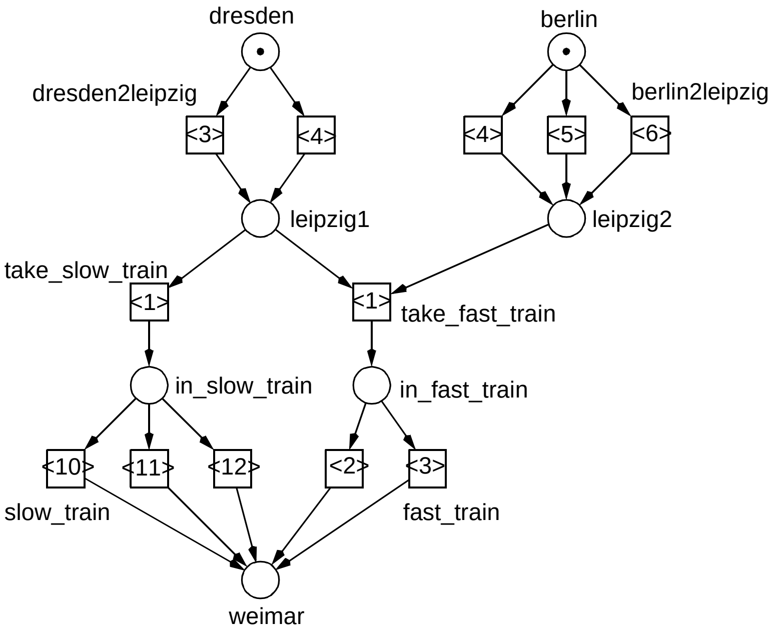


min_duration(dresden, weimar):

D net with lower bounds only:	14
DI net with lower and upper bounds:	7

-> maximal path by upper bounds only ? (!)

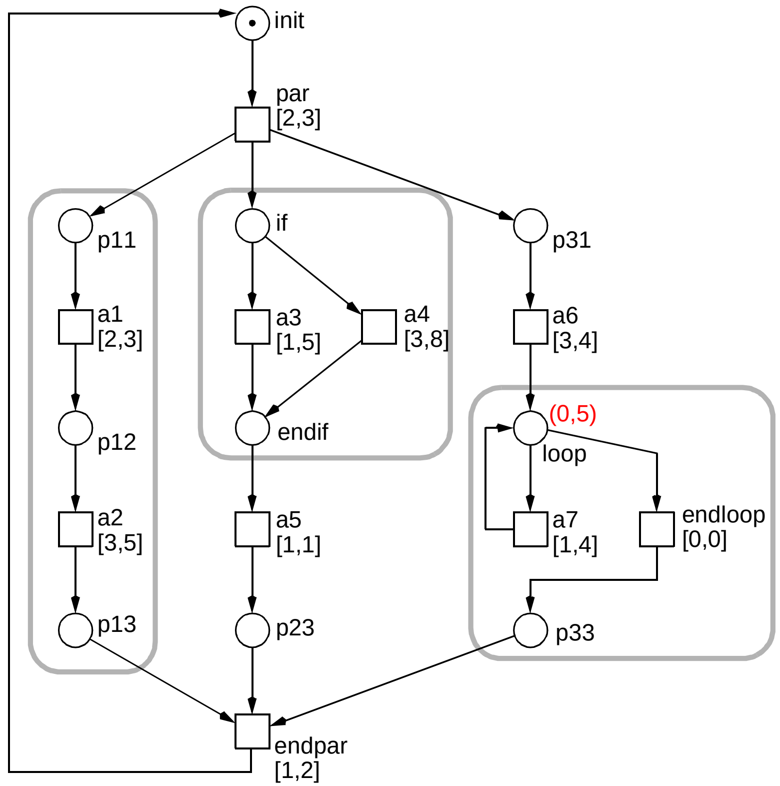
COUNTER EXAMPLE AS D NET



troublemaker

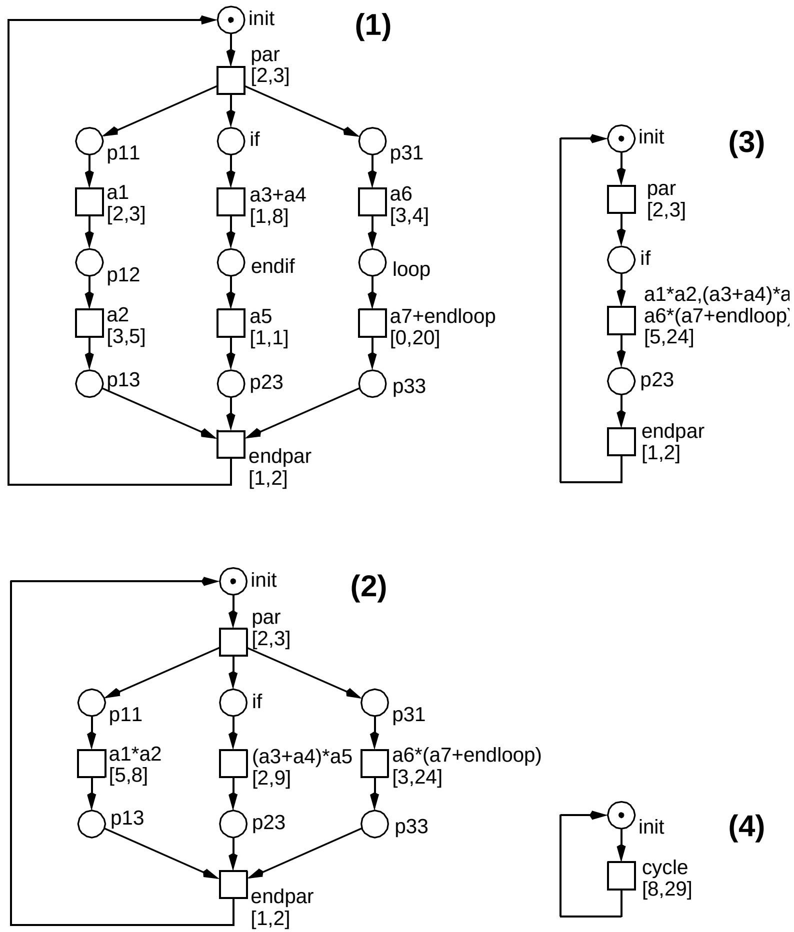
overlapping time windows of
dresden2leipzig & berlin2leipzig

STRUCTURAL COMPRESSION OF WELL-FORMED NET STRUCTURES, EXAMPLE

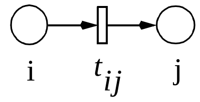


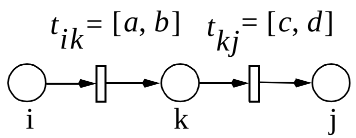
number of iterations

STRUCTURAL COMPRESSION OF WELL-FORMED NET STRUCTURES, EXAMPLE (CONT.)

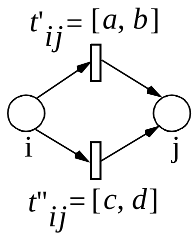


STRUCTURAL COMPRESSION OF WELL-FORMED NET STRUCTURES, GENERAL

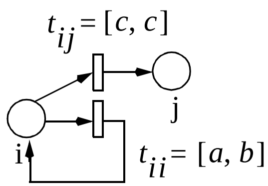




$$t_{ij} = [a + c, b + d]$$



$$t_{ij} = [min(a, c), max(b, d)]$$

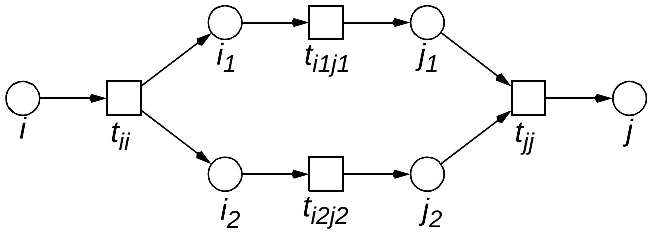


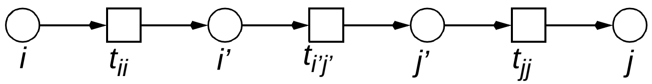
$$t_{ij} = [m \cdot a + c, n \cdot b + c]$$

assumption:

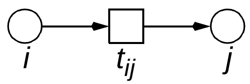
$\left\{\begin{matrix}\text{lower}\\ \text{upper}\end{matrix}\right\}$ bound $\left\{\begin{matrix}m\\ n\end{matrix}\right\}$ of iterations given

STRUCTURAL COMPRESSION OF WELL-FORMED NET STRUCTURES, GENERAL (CONT.)



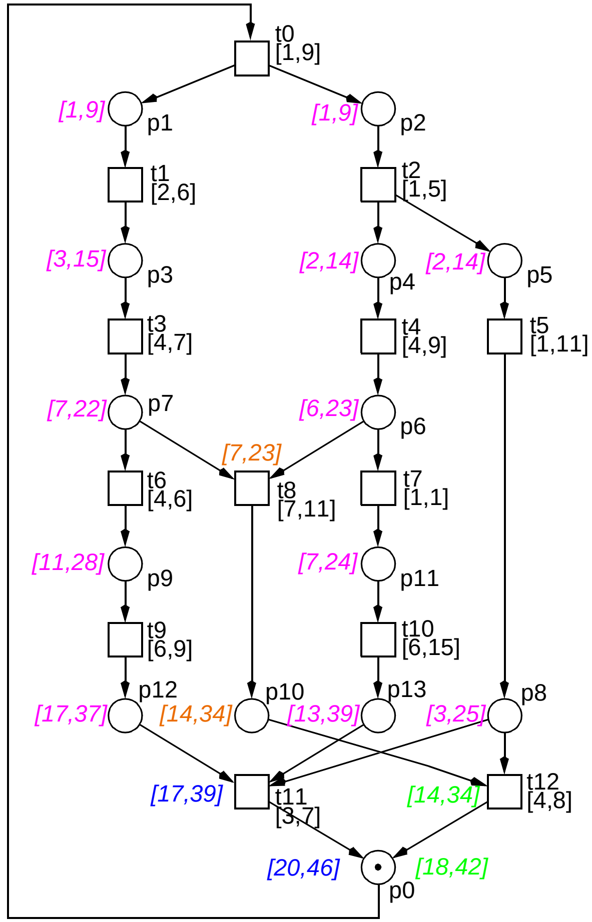


$$t_{i'j'} = [\ \max(\ low(t_{i1j1}),\ low(t_{i2j2})\ ),\ \max(\ upp(t_{i1j1}),\ upp(t_{i2j2})\ )]$$



$$t_{ij} = [\ low(t_{ii}) + low(t_{i'j'}) + low(t_{jj}),\ upp(t_{ii}) + upp(t_{i'j'}) + upp(t_{jj})]$$

EXAMPLE - INTERVAL EVALUATION OF NON-WELL-FORMED STRUCTURES, BUT 1-BOUNDED, ACYCLIC, ...



[Reske 95, p. 92]

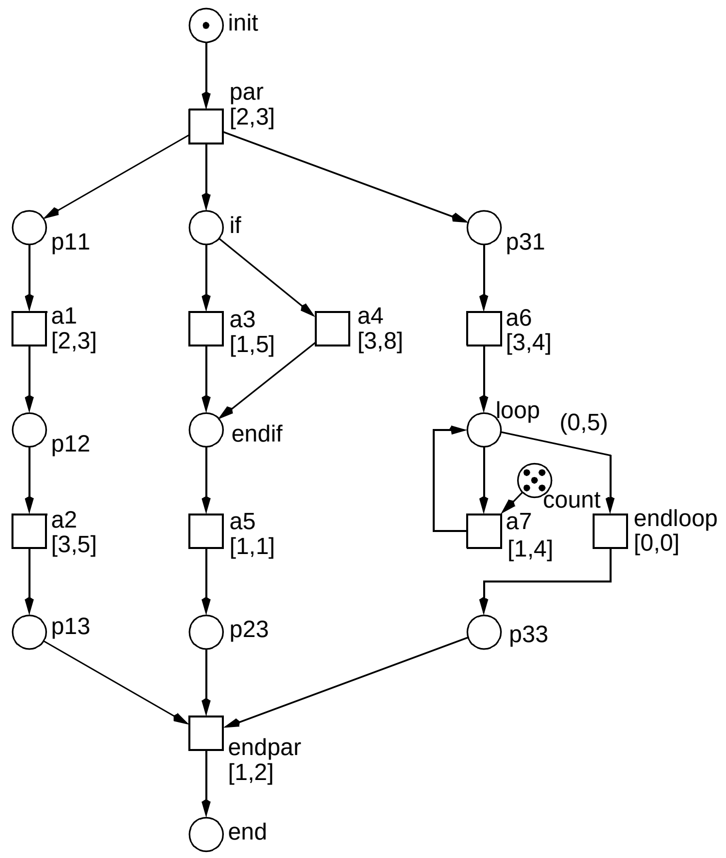
[18,46] cycle time

INTERVAL EVALUATION, GENERAL PROCEDURE

- net structure transformation
  - [first state -> init state]
  - [last state -> dead state]
  - resolution of (unlimited) cycles, if any
- net type transformation
  - DI net -> I net or
  - DI net -> D net;
- determine (set of) state numbers of
  - first state
  - last state

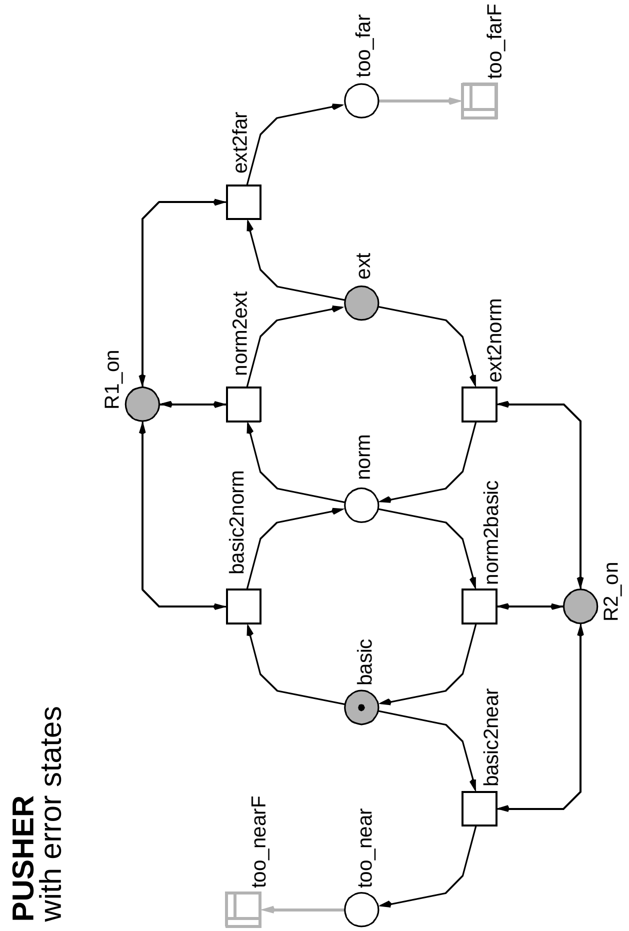
  of the path to be measured;
- evaluation of
  - reachability graph OR ?
  - other descriptions of all possible behaviours
    - prefix of branching processes
    - concurrent automaton

EXAMPLE OF NET STRUCTURE TRANSFORMATION

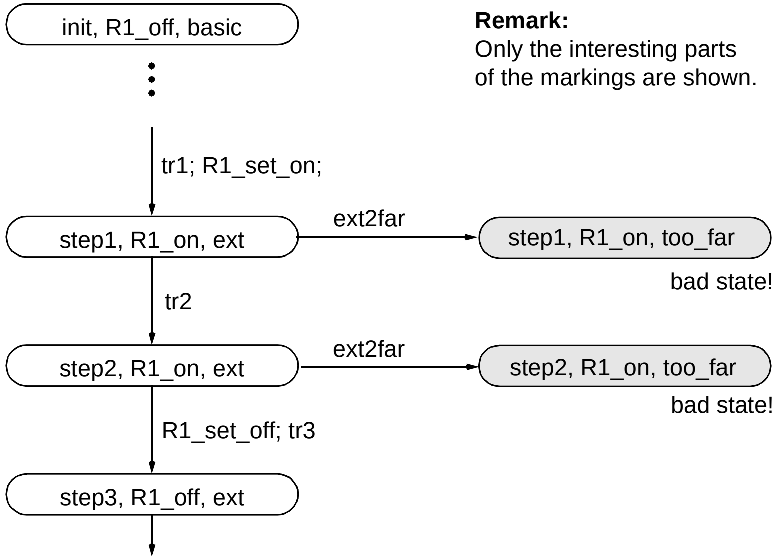


MIN (pathes(init, any dead state));
MAX (pathes(init, any dead state));

ENVIRONMENT MODEL, WITH EXPLICIT ERROR STATES



CONCURRENT PUSHERS, (PART OF THE) REACHABILITY GRAPH



Remark:
Only the interesting parts of the markings are shown.

-> (preemptive) interval nets

unreachability of bad states, m_0-dead(ext2far) if:

$$lft(tr2) < eft(ext2far) \wedge$$
$$lft(R1\_set\_off) < eft(ext2far)$$

REFERENCES

[Bause 96]
Bause; F.; Kritzinger, P. S.:
Stochastic Petri Nets, An Introduction to the Theory;
Vieweg 1996.

[Heiner 97a]
Heiner, M., Popova-Zeugmann, P.:
Worst-case Analysis of Concurrent Systems with Duration Interval Petri Nets;
Proc. 5. Fachtagung Entwurf komplexer Automatisierungssysteme (EKA ’97), Braunschweig, May 1997, pp. 162-179.

[Heiner 97b]
Heiner. M.; Popova-Zeugmann, L.:
On Integration of Qualitative and Quantitative Analysis of Manufacturing Systems Using Petri Nets;
Proc. 42. Int. wissenschaftliches Kolloquium (IWK ’97), Ilmenau, September 1997, Vol. 1, pp. 557-562.

[Merlin 74]
Merlin, P.:
A Study of the Recoverability of Communication Protocols;
PhD Thesis, Univ. of California, Computer Science Dep., Irvine, 1974.

[Popova 91]
Popova, L.:
On Time Petri Nets;
J. Information Processing and Cybernetics EIK 27(91)4, 227-244.

[Popova 94]
Popova-Zeugmann, L.:
On Time Invariance in Time Petri Nets;
Humboldt University at Berlin, Informatik-Bericht No. 39, December 1994.

[Popova 95]
Popova, L.:
On Liveness and Boundedness in Time Petri Nets;
Proc. CSP ‘95 Workshop, Warsaw, October 1995, pp. 136-145.

[Ramchandani 74]
Ramchandani, C.:
Analysis of Asynchronous Concurrrent Systems Using Petri Nets;
PhD Thesis, MIT, TR 120, Cambridge (Mass.), 1974.

[Starke 90]
Starke, P. H.:
Analysis of Petri Net Models (in German);
B.G.Teubner 1990.

[Starke 95]
Starke, P.:
A Memo On Time Constraints in Petri Nets;
Humboldt-University zu Berlin, Informatik-Bericht Nr. 46, August 1995.