(INFORMAL) INTRODUCTION INTO PETRI NETS ANALYSIS

QUALITATIVE ANALYIS METHODS, OVERVIEW

Static analysis:

- NET REDUCTION
- STRUCTURAL PROPERTIES
- LINEAR PROGRAMMING
  - place / transition invariants
  - state equation
  - trap equation

Dynamic analysis *(model checking)*:

- REACHABILITY ANALYSIS
  - (complete) reachability graph
  - compressed state spaces
    - BDDs, NDDs, ..., XDDs
    - Kronecker products
  - reduced state spaces
    - coverability graph
    - symmetry
    - stubborn sets
  - branching process

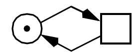
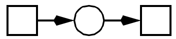
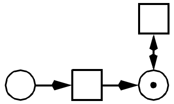
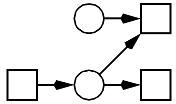
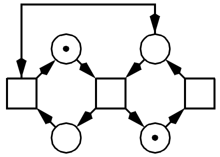
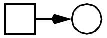
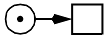
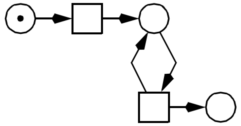
TYPICAL BASIC BEHAVIOURAL PETRI NET PROPERTIES

- How many tokens may reside at most in a given place . . .
  - -> (0, 1, k, oo) ?
    - **->** **BOUNDEDNESS**
- How often may a transition fire . . .
  - -> (0-times, 1-times, n-times, oo-times) ?
    - **->** **LIVENESS**
- Is the initial system state . . .
  - -> always reachable again ?
    - **->** **REVERSIBILITY**

ORTHOGONAL

- Is a given system state . . .
  - -> always reachable again ?
    - **->** **REPRODUCIBILITY** **->** **HOME STATE**
- Is a given system state
  - -> definitely reachable?
    - **->** **PROGRESS PROPERTIES**
  - -> never reachable ?
    - **->** **SAFETY PROPERTIES**

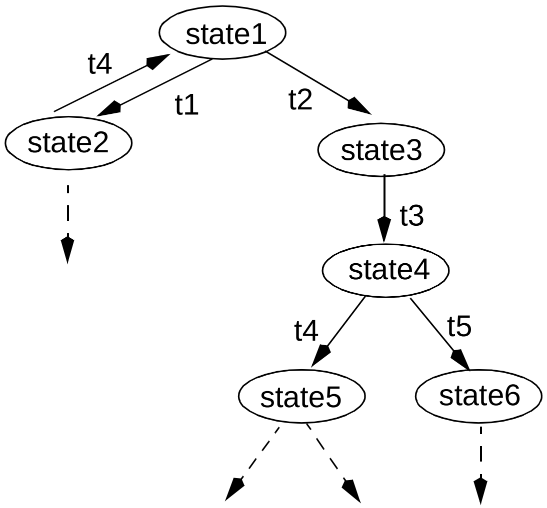
THREE ORTHOGONAL BEHAVIOURAL PROPERTIES

B L R 	**nB L R** 
B nL R 	**nB nL R** 
B L nR 	**nB L nR** 
B nL nR 	**nB nL nR** 

TYPICAL ANALYSIS TECHNIQUES (TO BE CONTINUED)

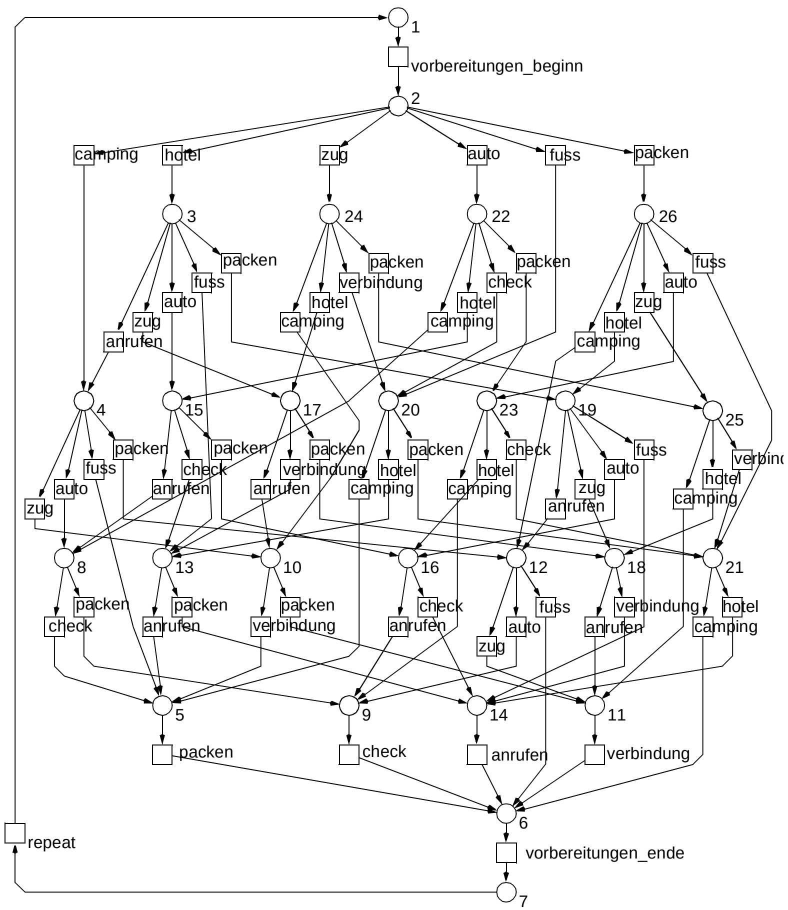
- **token game (?)**
- reachability graph (RG)

  nodes: system states
  arcs: the (single) firing transition



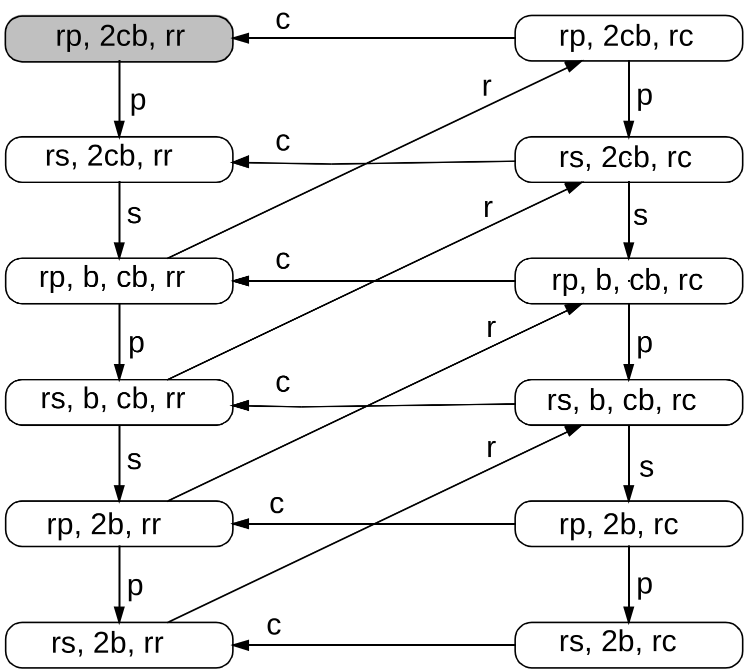
-> interleaving description of the whole system behavior

EX REACHABILITY GRAPH, TRAVEL PREPARATION



PUR	ORD	HOM	NBM	CSV	SCF	FT0	TF0	FP0	PF0	CON	SC	NC
N	Y	Y	Y	N	N	Y	Y	Y	Y	Y	Y	ES
RKTH	STP	CPI	CTI	SCTI	SB	k-B	1-B	DCF	DSt	DTr	LIV	REV
N	Y	Y	Y	Y	Y	1	Y	N	0	Y	Y	Y

EX REACHABILITY GRAPH, PRODUCER/CONSUMER, BOUNDED



buffer capacity = 2

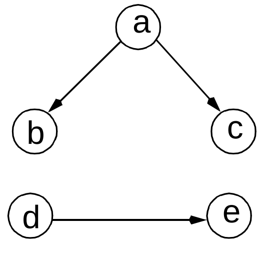
STRONGLY CONNECTED GRAPH

- basic graph properties

  -> applies also for general (monochromatic) graphs

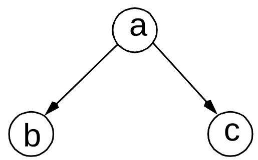
- needs directed graphs

  undirected graphs:
  connected = strongly connected



not connected

- for each pair of nodes a, b holds:
  there exists a path
  from a to b
  -> path(a, b);



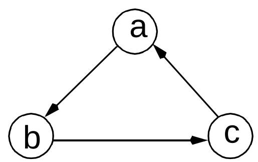
connected

- path(a, b):
  sequence of arcs
  starting at a and
  ending at b;

- general importance

  ex:
  system of one-way streets;

  question:
  is every place (intersection) from any place reachable?



strongly connected

EX: BASIC PROPERTIES AND REACHABILITY GRAPH

- ❑ no concurrency
  -> rg(pn) == pn

- ❑ rg - finite
  -> bounded pn

- ❑ rg - not sc
  -> pn not reversible

- ❑ no dead states,
  but liveness?

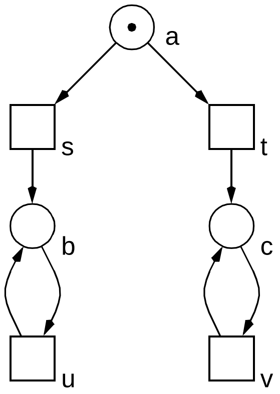
- ❑ condensed rg

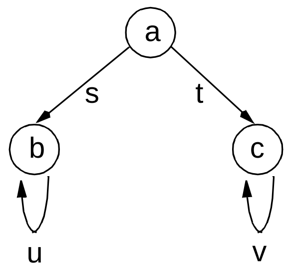
  node - sc component (scc)

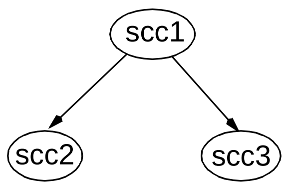
  scc:
  maximal set of sc nodes;

  a terminal scc
  -> *possible terminal system behavior*
  -> *must contain all transitions in a live pn*

- ❑ not all terminal scc contain all transitions
  -> the pn is not live







BASIC PROPERTIES AND REACHABILITY GRAPH

- ❑ How many tokens may reside at most in a given place . . .
  -> (0, 1, k, oo) ?
  -> BOUNDEDNESS
  -> RG IS FINITE

- ❑ How often may a transition fire . . .
  -> (0-times, n-times, oo-times) ?
  -> LIVENESS
  -> EVERY TERMINAL SCC CONTAINS ALL TRANSITIONS

- ❑ Is the initial system state . . .
  -> always reachable again ?
  -> REVERSIBILITY
  -> RG IS SC (CONSISTS OF ONE SCC)

- ❑ home state

SOFTWARE-ORIENTED INTERPRETATION OF NET PROPERTIES

- **Dead code**
  statements/actions which will never be executed;
  pn: *the corresponding transition never fires (dead at the initial marking);*
  rg: *transition does not appear at any edge;*

- **Total deadlock**
  system state from which there is no exit;
  pn: *dead marking;*
  rg: *terminal nodes (no outgoing arcs);*

- **Partial deadlock**
  not all parts of the system are available for all times;
  pn: *no dead markings, but dead transition(s);*
  rg: *not all terminal strongly connected components contain all transitions;*

SOFTWARE-ORIENTED INTERPRETATION OF NET PROPERTIES (CONT.)

- **Well-structuredness**
  all parts of the system may be executed for ever;
  pn: *the net ist live;*
  rg: *all terminal strongly connected components contain all transitions;*

- **Livelock**
  parts of the system may be blocked for ever (due to the scheduler‘s strategy or something else not contained in the model);
  pn: *live, but not livelock-free;*
  rg: *not all circles contain all transitions;*

- **Fault tolerance and self-synchronization**
  after a failure or from any abnormal state, the software will return to normal execution (recovery from failure) within finite time*;*
  pn: *reproducibility / reversibility;*
  rg: *from any state, the home state (initial state) is reachable again;*

REACHABILITY GRAPH, CONSTRUCTION ALGORITHM

PROCEDURE rg (**IN** Net *pn*, **IN** Marking *m0,*
OUT MSet *nodes*, **OUT** ArcSet *arcs*);

MSet $U = \{m0\}$, // unprocessed markings
$N = \{m0\}$; // rg nodes
ArcSet $E = \emptyset$; // rg arcs (pre, post, t)
Marking m'; // successor marking
Transition t;

WHILE $U \neq \emptyset$ **DO**
choose one $m \in U$;
$U = U - \{m\}$;

FOR ALL t enabled at m **DO**
$m' = m + \Delta t;$
IF $m' \notin N$ // new marking
THEN $N = N \cup \{m'\}$;
$U = U \cup \{m'\}$
ENDIF;
$E = E \cup \{(m, m', t)\}$
ENDFOR

ENDWHILE;

nodes = *N*; *arcs* = *E*;
ENDPROC rg.

REACHABILITY GRAPH, OBSERVATIONS

- **unbounded** Petri net
  -> the rg is **infinite**

  bounded Petri net
  -> the rg is **finite**

- simple construction algorithm
  -> single step firing rule

- concurrency
  -> enumeration of all interleaving sequences

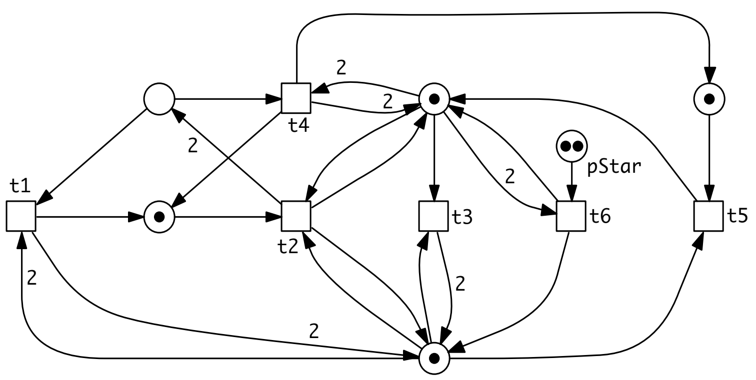
- branching arcs in the rg
  -> conflict **OR**
  -> concurrency

- rg tend to be very large
  -> automatic evaluation necessary

- worst case: **growth can not be bounded by primitive recursive function**

  -> alternative analyses techniques ?

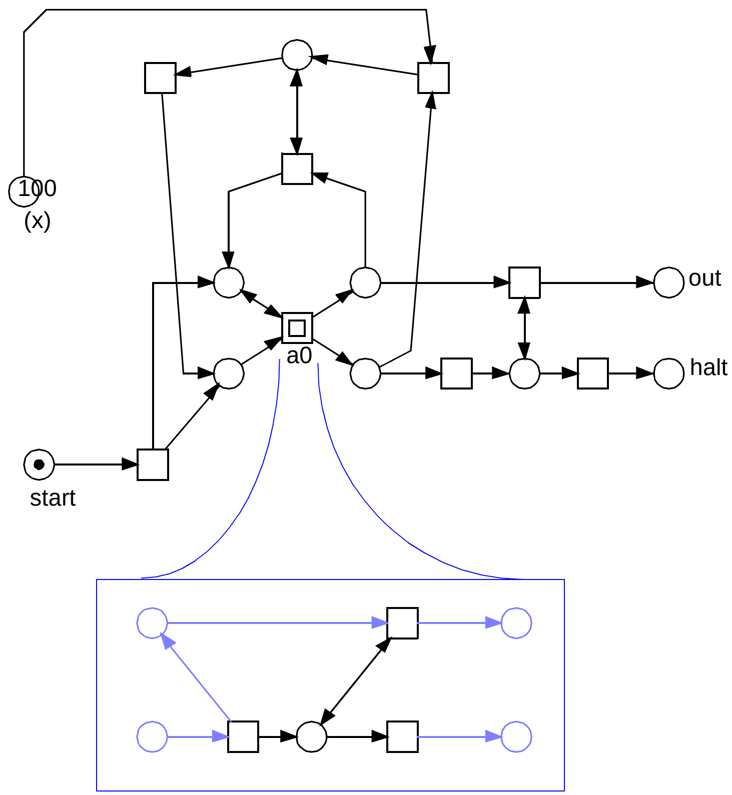
Ex: Jantzen



[Jantzen, PN newsletter 1983, p.25]
[Starke 1990, p.51]

```
k = m0(pStar)
Z(k) = maximal token number in a reachable marking
M(k) = number of reachable markings
-----
-----
Z(k) = 2*f(k) + 2
f(0) = 2
f(k+1) = f(k)*2^{f(k)}
------
Z(0) = 6                                   M(0) = 30
Z(1) = 18                                  M(1) = 427
Z(2) = 4098
Z(3) = 2^2060 + 2
```


Ex: Ackermann



Ackermann function - not primitive recursive

a pn, weakly calculating A_{n+1} (x)

[Priese, Wimmel 2003, p.141]

Petri net properties, Overview / Charlie

1. Simple Structural Properties

PUR	pure *(no side conditions)*
ORD	ordinary *(1-multiplicity of all arcs)*
HOM	homogeneous *(all output arcs of a given place have the same multiplicity)*
NBM	non-blocking multiplicity *(for each place applies: MIN multiplicity of input arcs >= MAX multiplicity of output arcs)*
CSV	conservative *(any firing preserves token amount)*
SCF	static conflict free
Ft0	every transition has a pre-place
tF0	every transition has a post-place
FP0	every place has a pre-transition
pF0	every place has a post-transition
CON	connected
SC	strongly connected
MG	marked graph *(synchronization graph)*
SM	state machine
FC	free choice net
EFC	extended free choice net
ES	extended simple net

More Expensive Structural Properties

2. Structural Properties

RKTH	rank theorem
STP	siphon trap property
SMC	state machine coverable *(covered with SM components)*
SMD	state machine decomposable *(covered with SCSM components)*
SMA	state machine allocatable
CPI	covered with place invariants
CTI	covered with transition invariants
SCTI	strongly covered by transition invariants
SB	structurally bounded

3. Behavioural Properties

k-B	k-bounded
DCF	dynamically conflict free
DSt	dead states *(a state where no transition is enabled)*
DTr	dead transitions *(at the initial state)*
LIV	live
REV	reversible *(the initial state m_0 can be reached again from all reachable states: home state)*