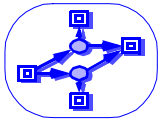
ABOUT SOME APPLICATIONS OF PETRI NET THEORY

(MY PETRI NET PICTURE BOOK)

MONIKA HEINER

BTU Cottbus
Computer Science Institute



C. A. PETRI - INTERPRETATIONS OF NET THEORY
GMD, INTERNAL REPORT 75-07, 2ND IMPROVED EDITION 1976

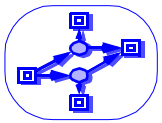
places	transitions
state elements	transitional elements
conditions	events/facts
statements	dependencies
model domains	specifications
chemical compounds	chemical reactions
open one-point sets	closed one-point sets
channels	offices
languages	translators
products	poduction activities

CONTENTS

PETRI NETS AS VEHICLE FOR REASONING ABOUT

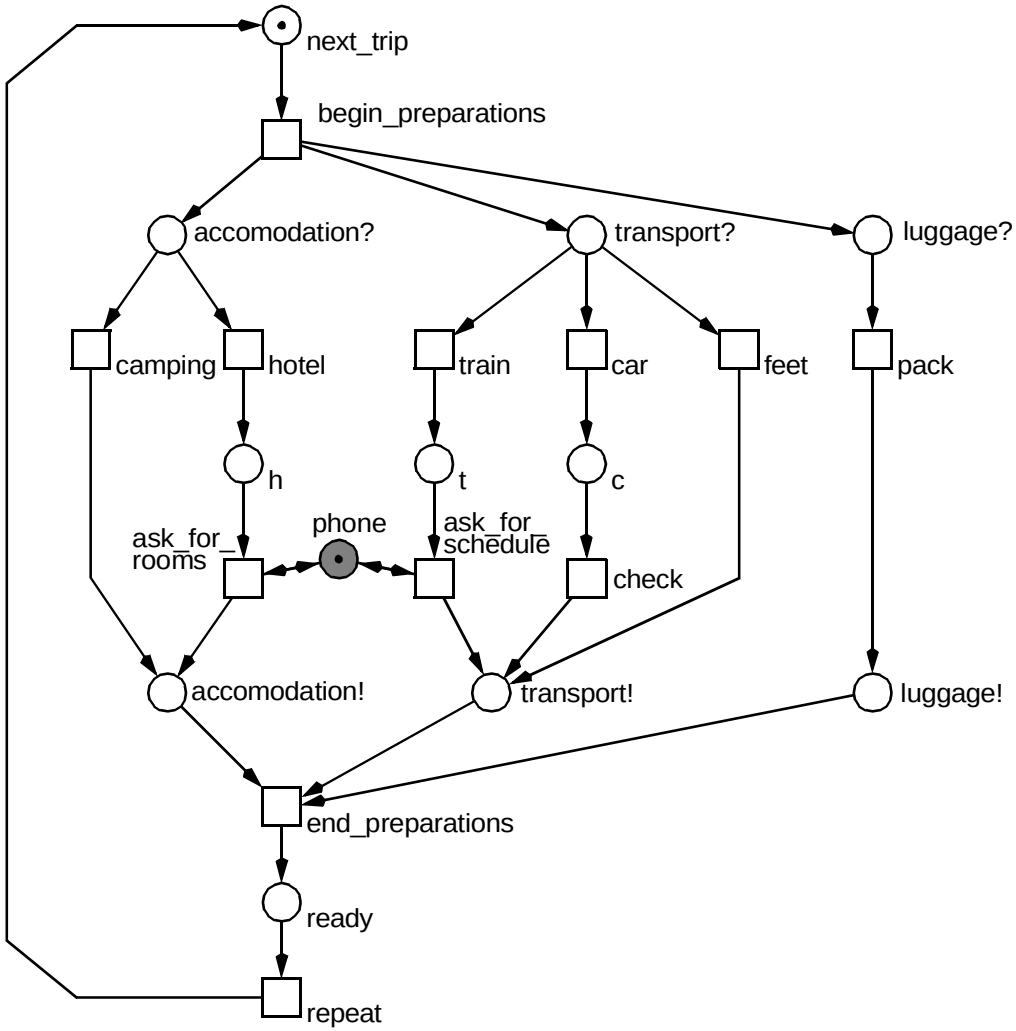
- administration patterns (work flows, process models)
- software structures
- hardware structures -> fault trees
- control engineering, to be added
- process/chemical engineering, to be added
- knowledge representation, to be added
- games
- biochemical networks (systems biology)
- music notations
- aesthetics

more . . . ?



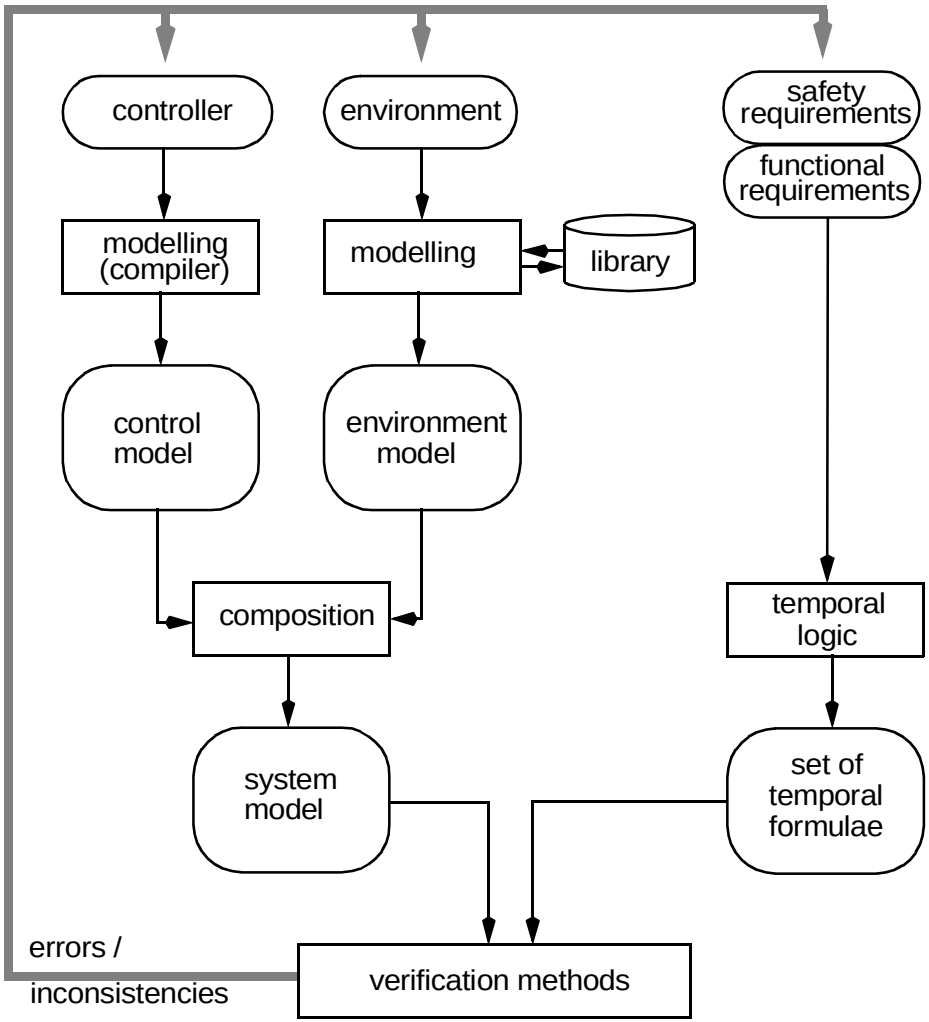
ADMINISTRATION, EXAMPLE1

TRAVEL PREPARATION



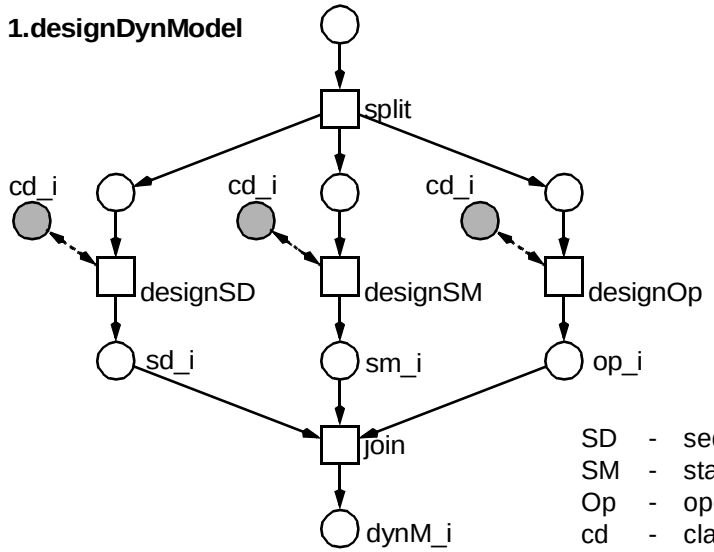
ADMINISTRATION, EXAMPLE2

SYSTEM VALIDATION, PROCESS AND TOOLS

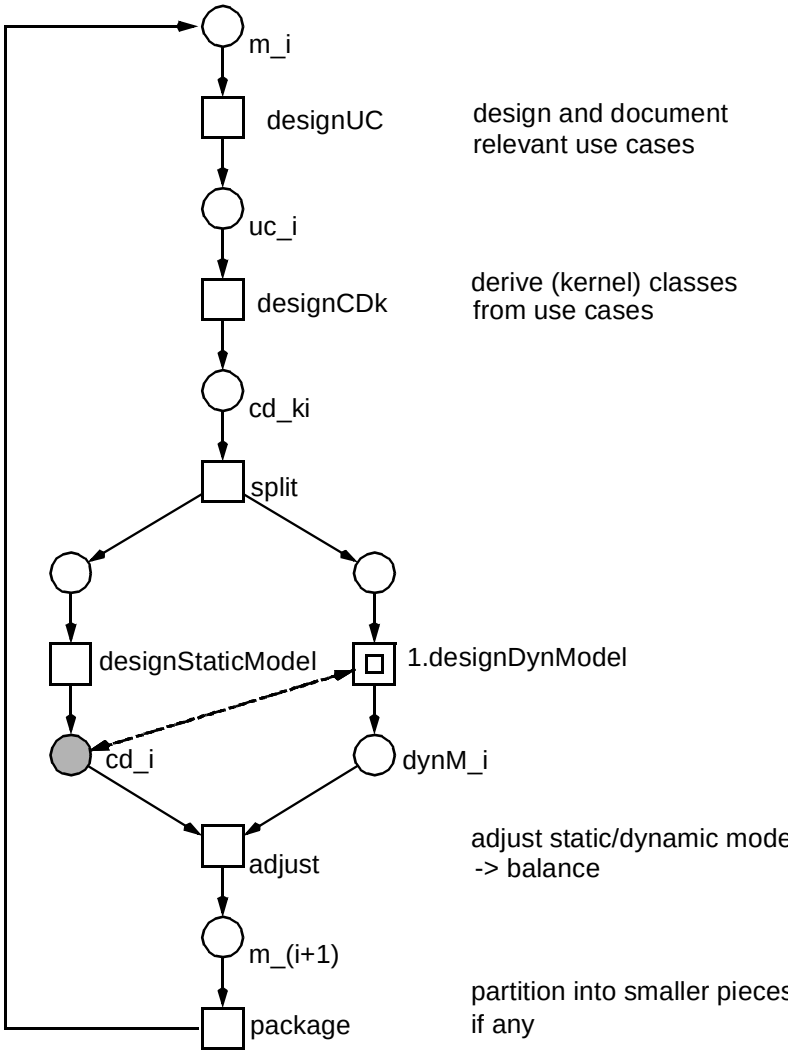


UML MODELLING, MACRO PROCESS

- [Balzert 2001, 386-391]



SD - sequence diagram
SM - state machine (state charts)
Op - operations
cd - class diagram



ADMINISTRATION, SUMMARY

PLACES

- ❑

TRANSITIONS

- ❑

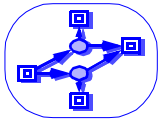
FLOW ARCS

- ❑

TOKENS

- ❑

HOME WORK

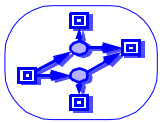


SOFTWARE, EXAMPLE1A

SEQUENTIAL PROGRAM

- first all-zero row in a given matrix
- source code, strongly structured
- no goto‘s, but goto walk-arounds

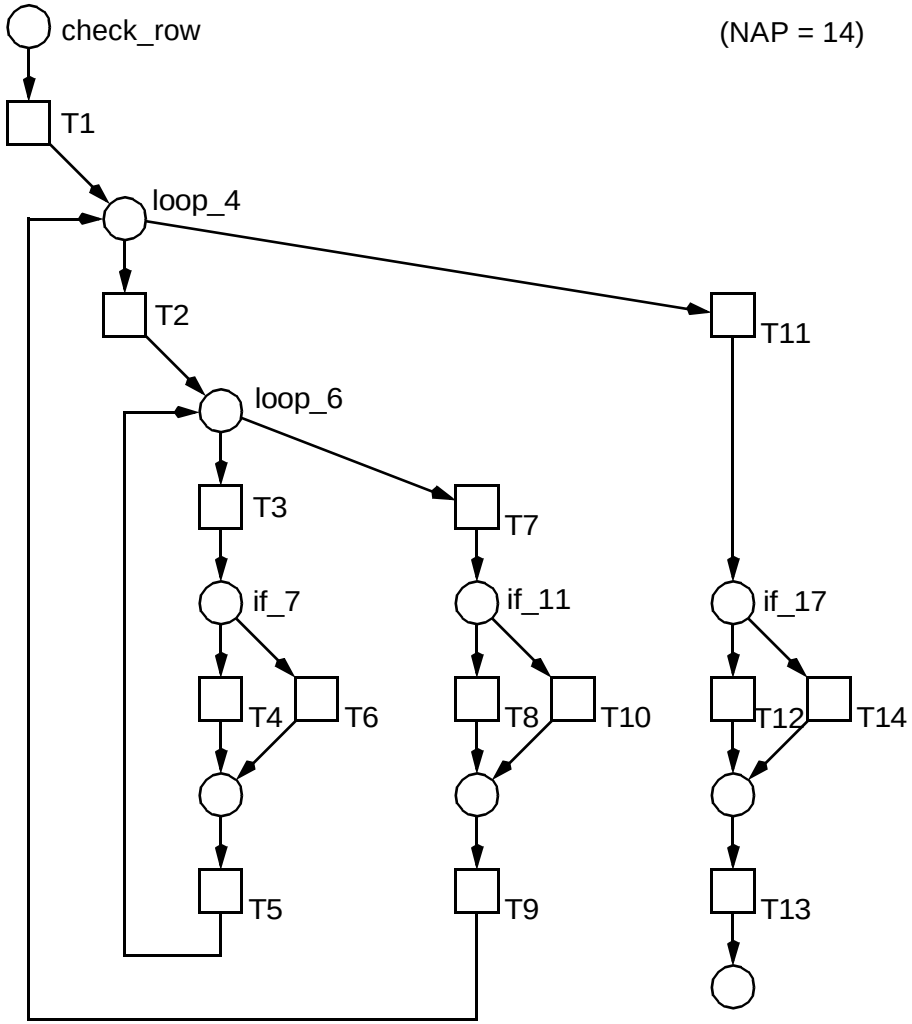
```
1   Boolean
2       no_row_found := true,
3       all_zero;
4   check_row: loop (all i and no_row_found)
5       all_zero := true;
6       check_column: loop (all j and all_zero)
7           if x[i,j] ≠ 0
8           then all_zero := false              ! abnormal termination of
9           endif                               ! check_column loop
10      endloop check_column;
11      if all_zero                             ! normal termination of
12      then                                    ! check_column loop
13          write(i);
14          no_row_found := false               ! abnormal termination of
15      endif                                   ! check_row loop
16  endloop check_row;
17  if no_row_found                             ! normal termination of
18  then                                        ! check_row loop
19      write(“no”)
20  endif
```



SOFTWARE, EXAMPLE1A

SEQUENTIAL PROGRAM

- control structure model
- abstraction of data-dependent branching
- Number of Acyclic Paths (NAP): 14



SOFTWARE, EXAMPLE1B

SEQUENTIAL PROGRAM

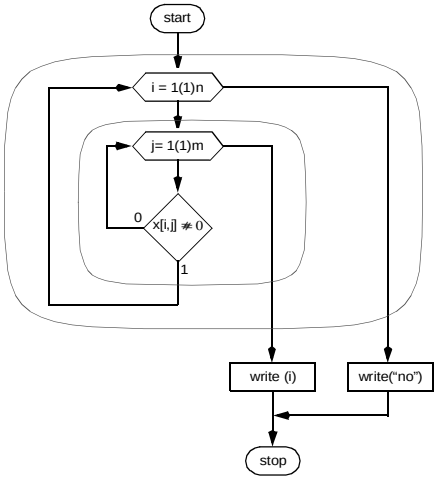
- source code, well-structured
- structured goto‘s only

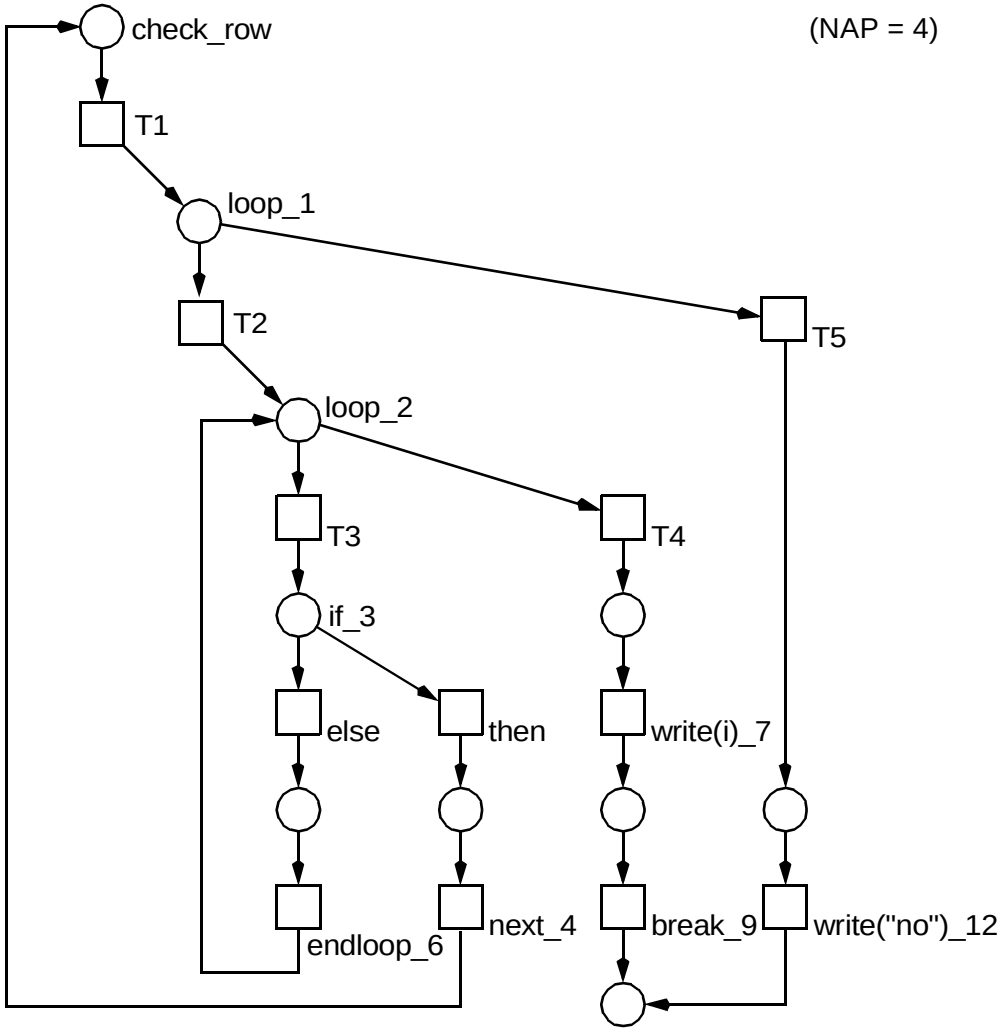
```
1   check_row: loop all i
2         check_column: loop all j
3               if x[i,j] ≠ 0
4               then next check_row          ! abnormal termination of
5               endif                        ! check_column loop
6         endloop check_column;
7         write(i);                          ! normal termination of
8                                            ! check_column loop
9         break;                             ! abnormal termination of
10                                           ! check_row loop
11  endloop check_row;
12  write(“no”)                              ! normal termination of
                                             ! check_row loop
```

SOFTWARE, EXAMPLE1B

SEQUENTIAL PROGRAM

- control structure model
- NAP: 4

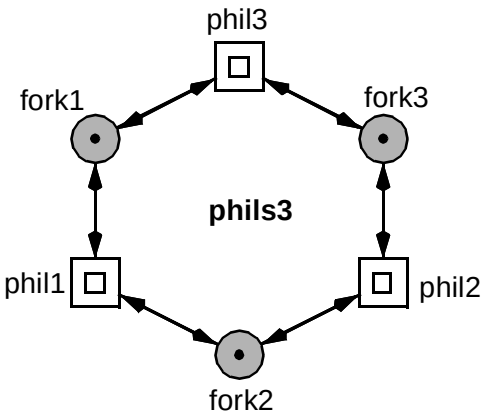


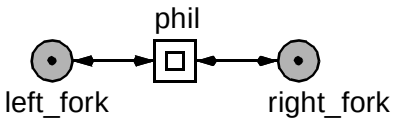


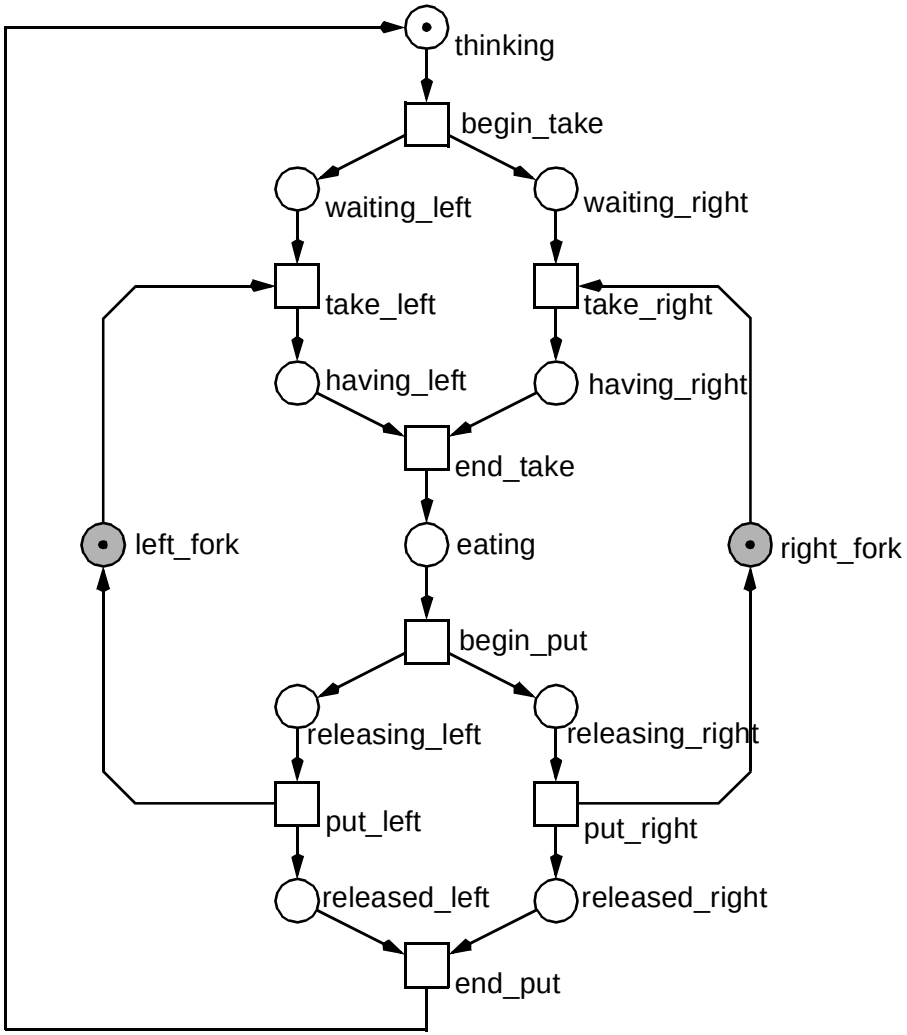
SOFTWARE, EXAMPLE2

CONCURRENT PROGRAM

- dining philosophers
- scalable -> benchmark for analysis algorithms



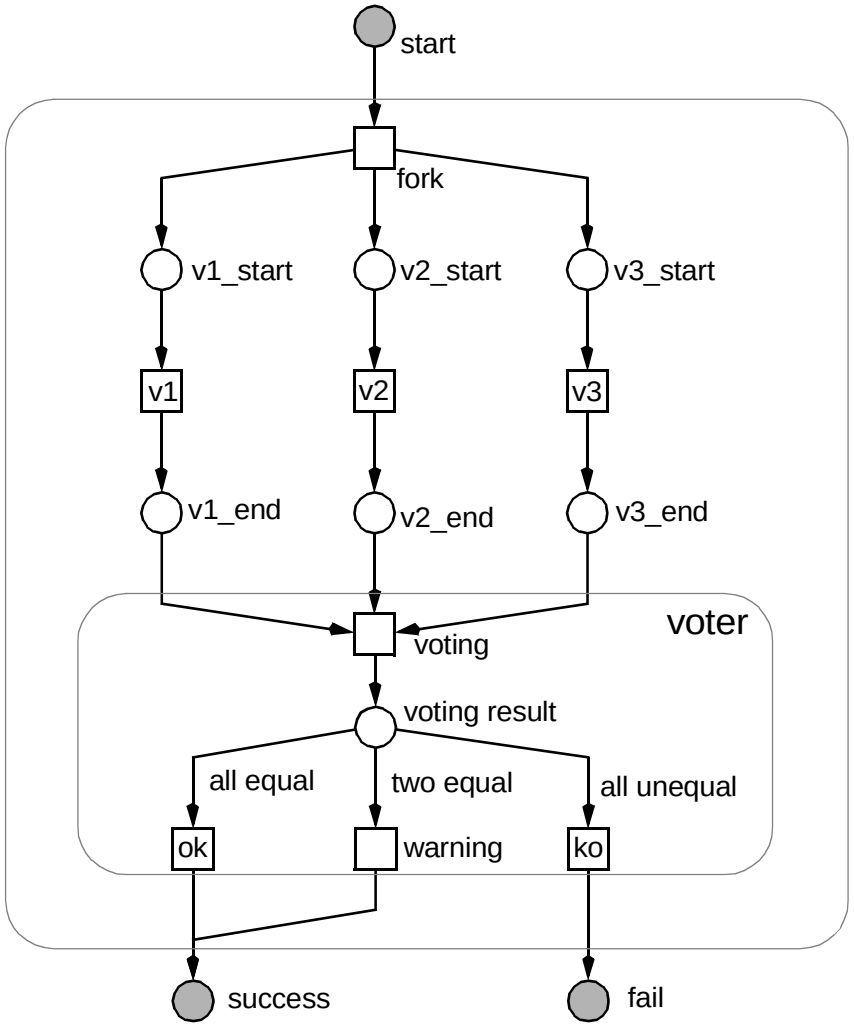




SOFTWARE, EXAMPLE3A

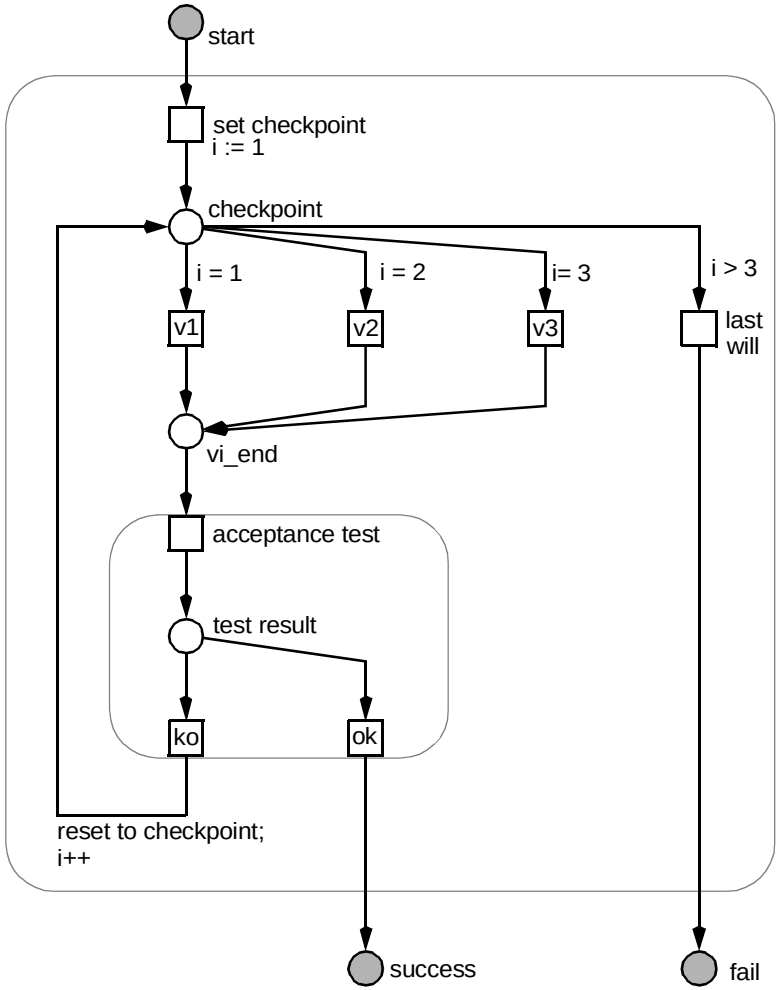
N VERSION PROGRAMMING

- parallel execution of n program versions
- followed by majority test
- higher abstraction level, transitions:
  -> program versions
  -> voting algorithm



RECOVERY BLOCK SCHEME

- alternative execution of n program versions
- each followed by acceptance test
- high-level Petri net



SOFTWARE, SUMMARY

PLACES

- control flow point

TRANSITIONS

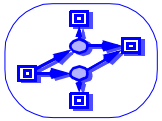
- statement, statement sequence, black-box algorithm

FLOW ARCS

- control flow, usually without branching conditions
- data flow = control flow + control variables

TOKENS

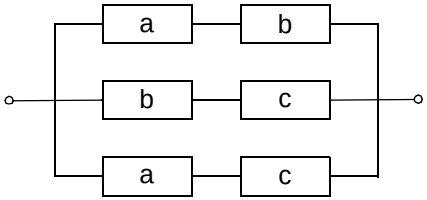
- execution counter
- synchronization/communication objects

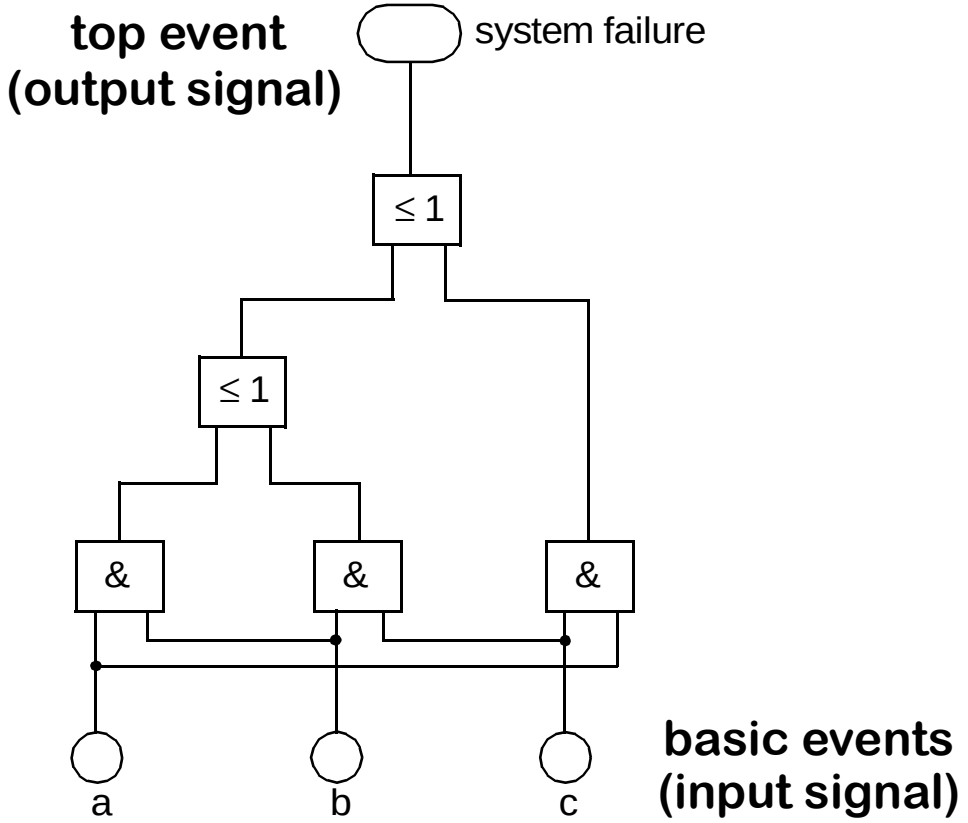


HARDWARE, EXAMPLE

FAULT TREES

- network notation of Boolean functions
- two gates: OR, AND
- multiple use of basic events allowed -> no real trees
- 2-of-3 system

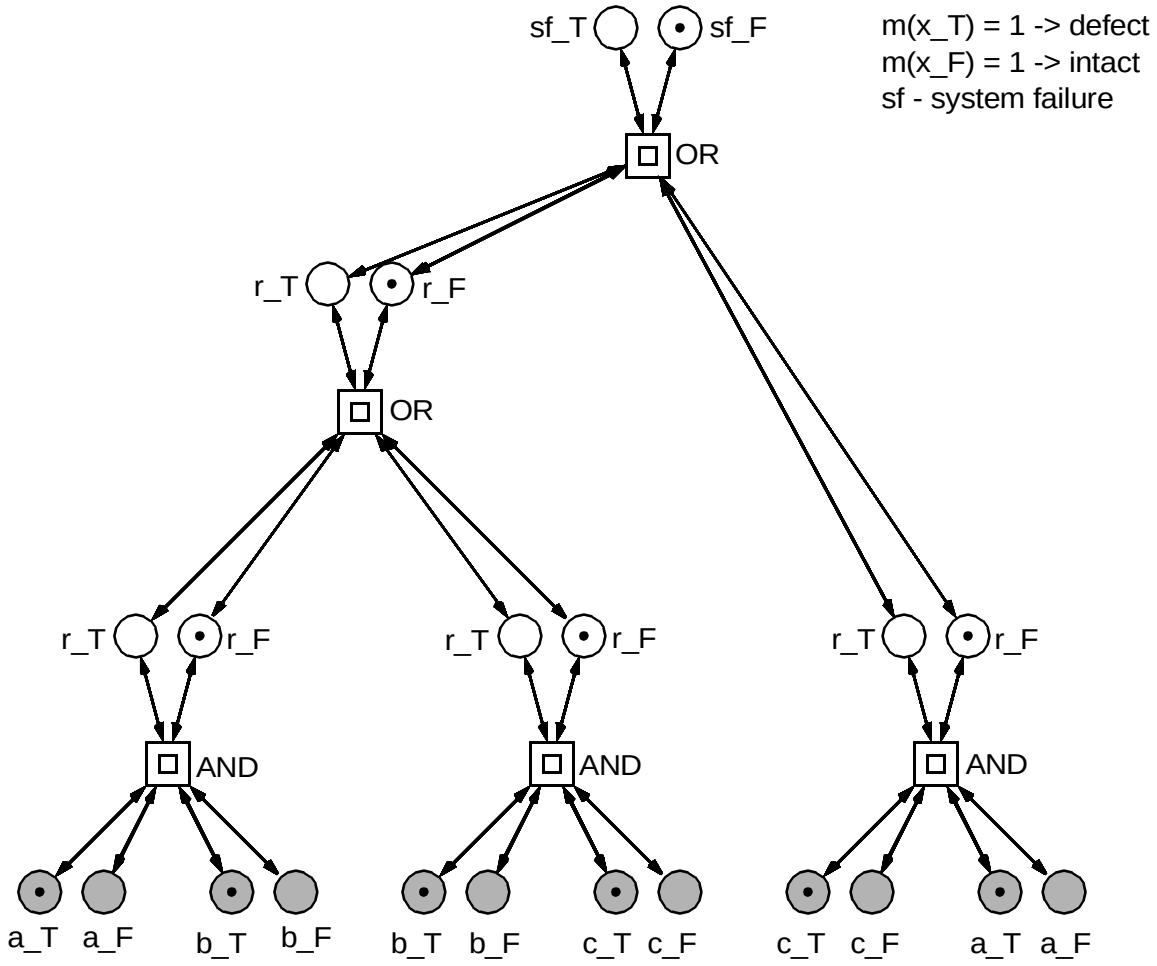




HARDWARE, EXAMPLE

FAULT TREES

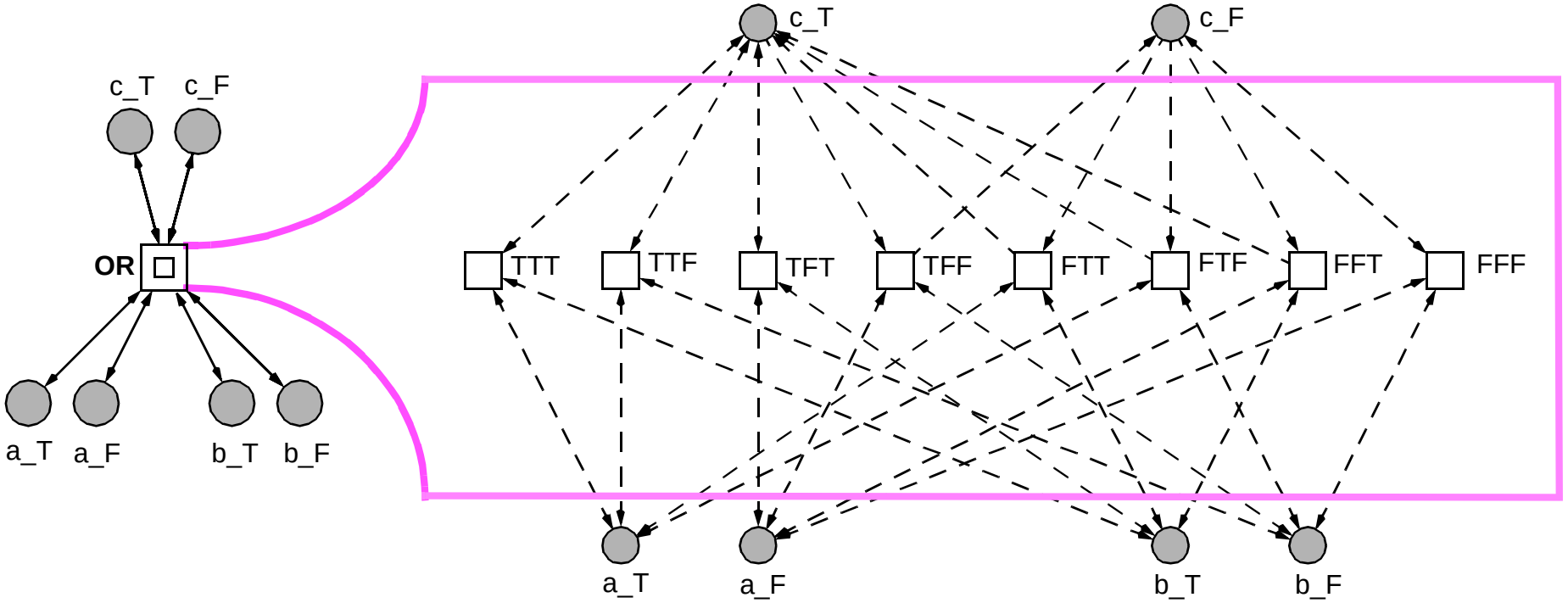
- states of a signal x:
  -> two places x_T, x_F
  -> 1-P-invariant
- input changes of
  basic events
  propagate to the top
- no dead states
- in 'stable state',
  the top event
  (system failure: T/F)
  is reproduced forever



HARDWARE, EXAMPLE

FAULT TREES

- OR gate
- persistent



HARDWARE, SUMMARY

PLACES

- current signal, two places each for the two possible states - on/off

TRANSITIONS

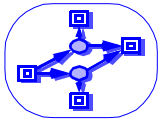
- logical gates

FLOW ARCS

- 'wires'

TOKENS

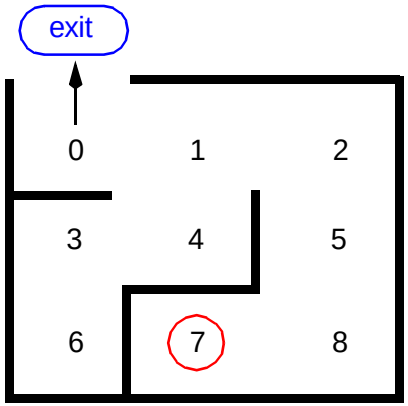
- on/off (high/low) signal

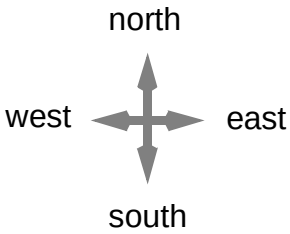


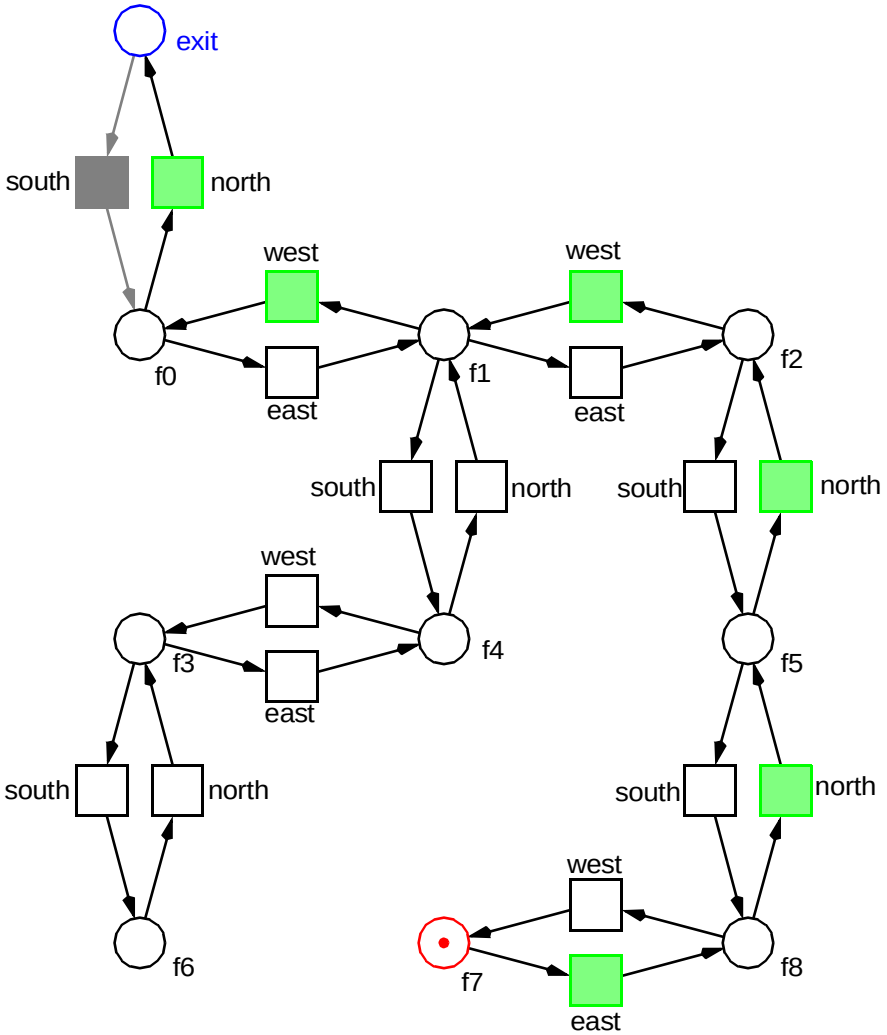
GAMES, EXAMPLE1

MAZE

- starting at any square, find the shortest path out of the maze
- generic pn construction
- deadlock analysis
  + shortest path to dead state





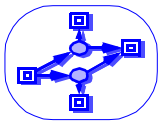


GAMES, EXAMPLE2

SOLITAIRE

- two versions,
  green squares Y/N
- all but one squares
  carry tokens
- remove tokens
  by jumbing over them
- goal of the game:
  only one token left
- questions:
  is there a solution ?
- always ?

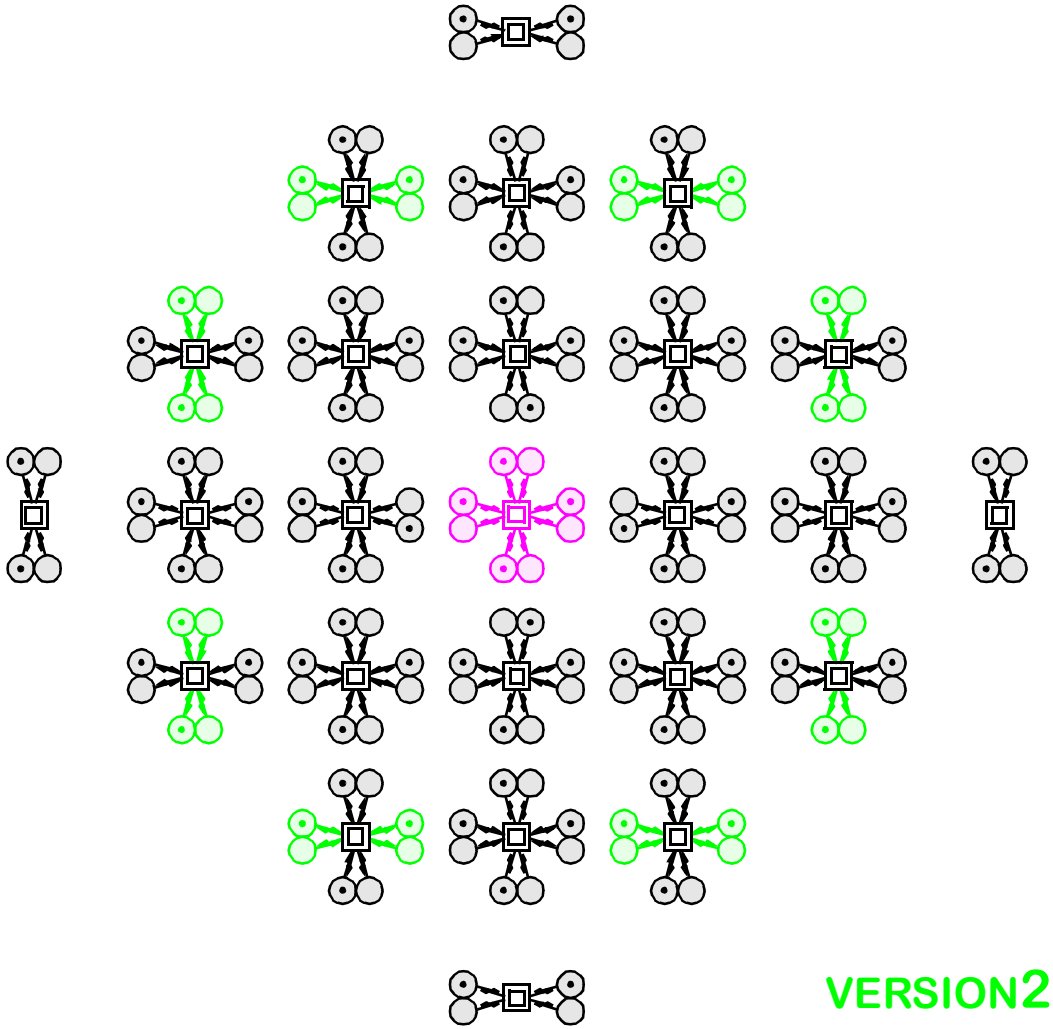
11	12	13	14	15	16	17
21	22	23	24	25	26	27
31	32	33	34	35	36	37
41	42	43	44	45	46	47
51	52	53	54	55	56	57
61	62	63	64	65	66	67
71	72	73	74	75	76	77



GAMES, EXAMPLE2

SOLITAIRE

- two-level hierarchical pn
- only one square net component
- two states for each square i: T(i), F(i)
- goal of the game: dead state(s) with Σ T(i) = 1
- reachable ?
- for any initial marking ?

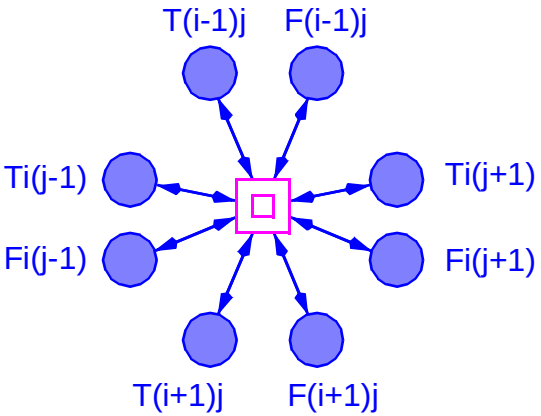


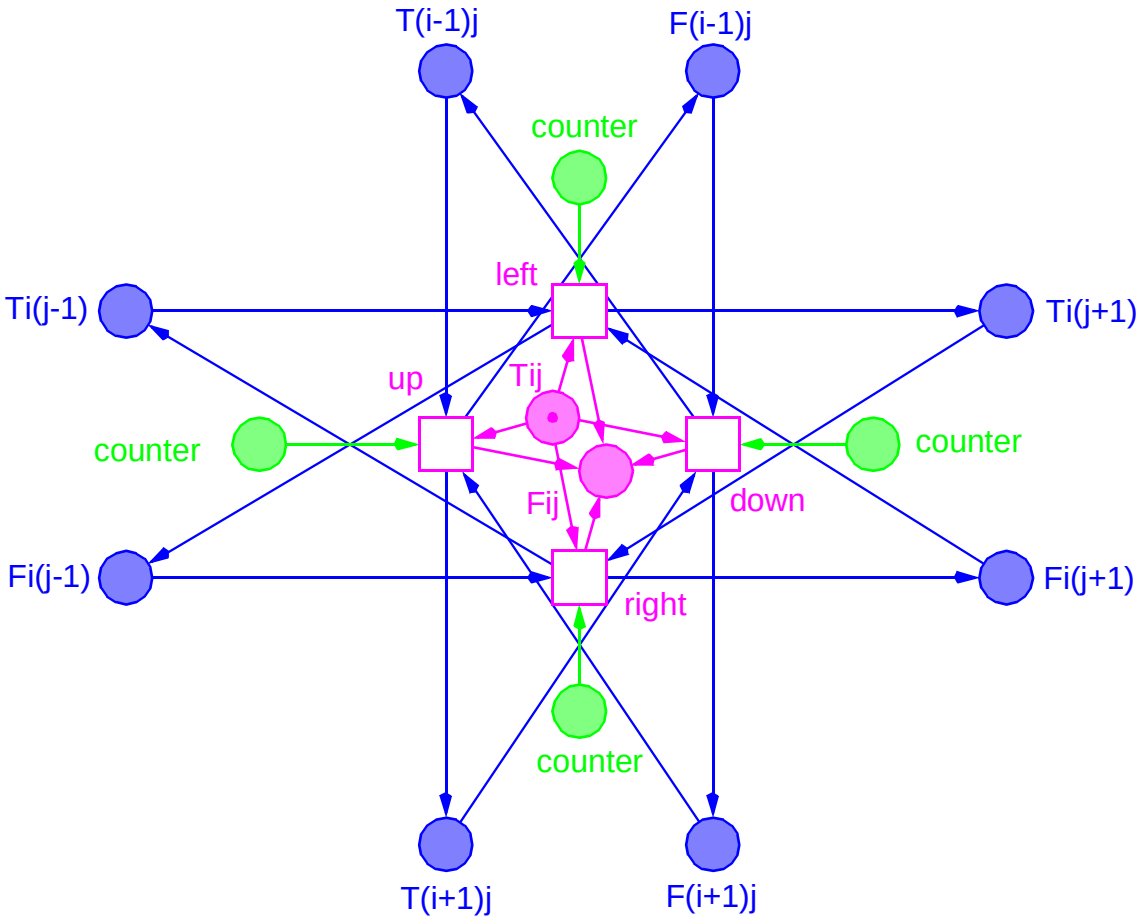
VERSION2

GAMES, EXAMPLE2

SOLITAIRE

- square component
- counter facilitates reachbility question, but hinders analysis





GAMES, SUMMARY

PLACES

- local game situation

TRANSITIONS

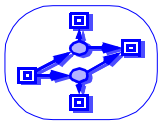
- moves of the game

FLOW ARCS

- changes in game situation

TOKENS

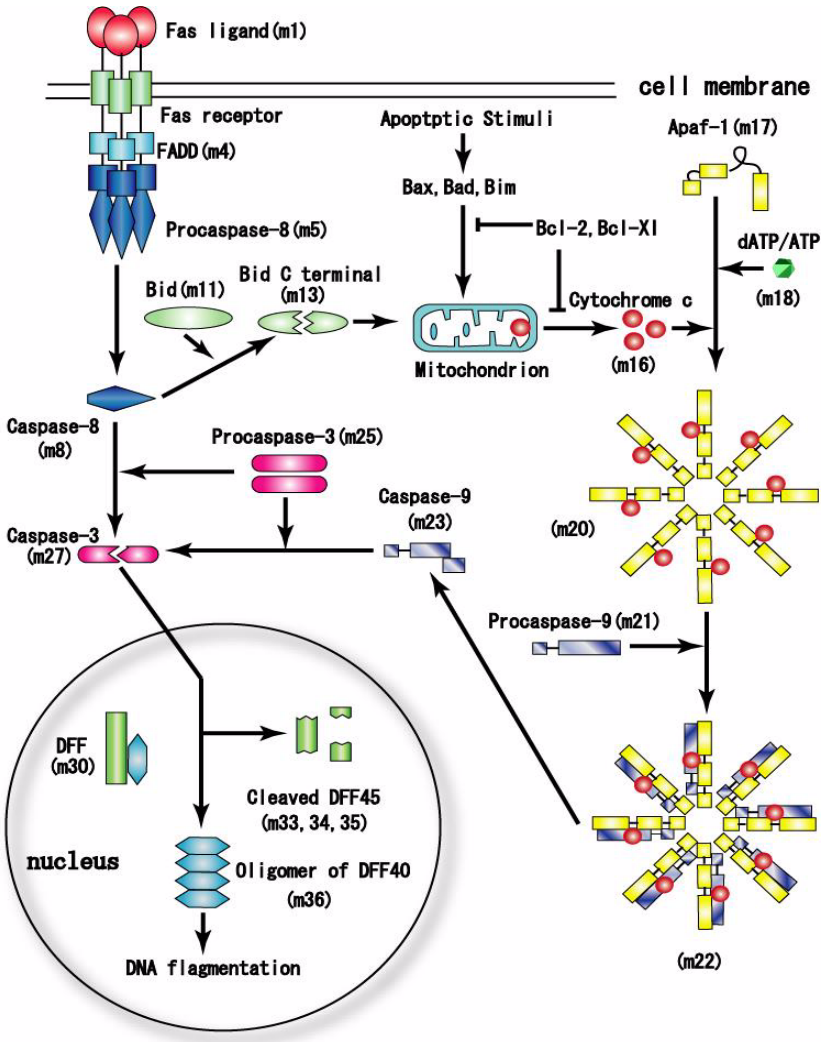
- game tokens or their negation



BIOCHEMISTRY, EXAMPLE1

APOPTOSIS

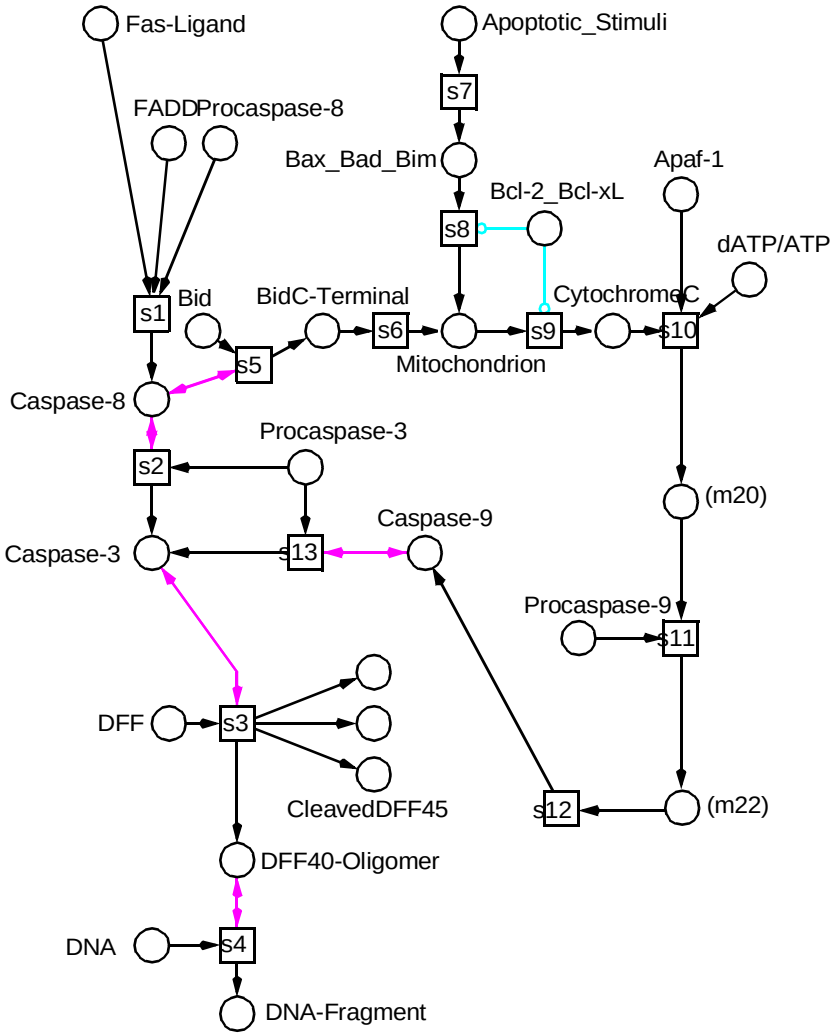
- signal transduction pathway
- general scheme
- inhibition



BIOCHEMISTRY, EXAMPLE1

APOPTOSIS

- ordinary Petri net
- signal transduction
  -> read arcs
- inhibition
  -> inhibition arcs
- analysis
  -> environment model
  -> read/inhibitor arcs



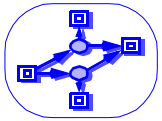
BIOCHEMISTRY, EXAMPLE2

POTATO TUBER

- metabolic pathway
- stoichiometric equations
- reversible reactions

sucrose ⟹ starch

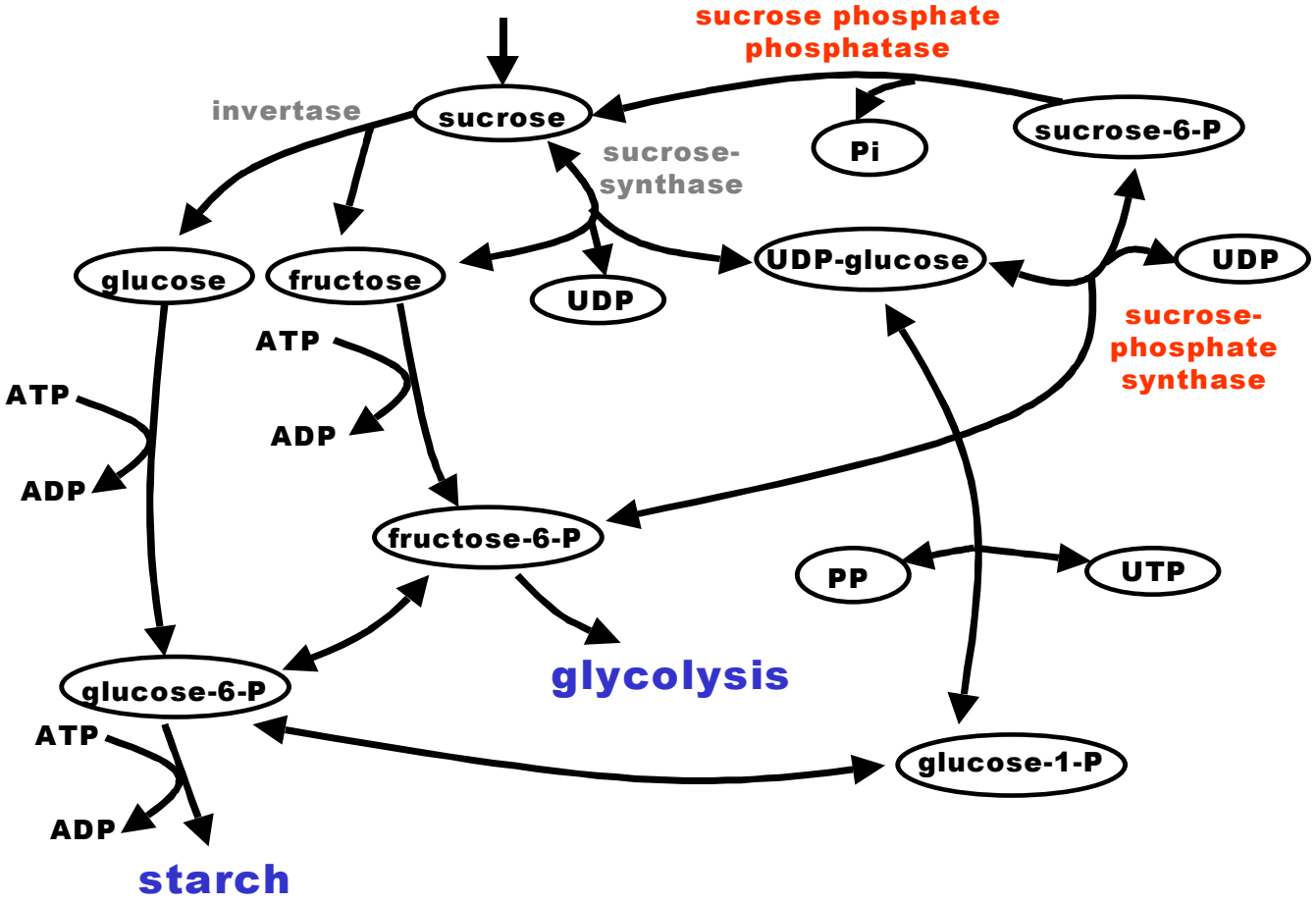
R1. sucrose synthase:	Suc + UDP <—> UDPglc + Frc
R2. UDPglucose pyrophosphorylase:	UDPglc + PP —> G1P + UTP
R3. phosphoglucomutase:	G1P <—> G6P
R4. fructokinase:	Frc + ATP —> F6P + ADP
R5. phosphoglucose isomerase:	F6P <—> G6P
R6. hexokinase:	Glc + ATP —> G6P +ADP
R7. invertase:	Suc —> Glc + Frc
R8. glycolysis:	F6P + 29 ADP + 28 Pi —> 29 ATP
R9. sucrose phospahate synthase:	F6P + UDPglc <—> S6P + UDP
R10. sucrose phosphate phosphatase:	S6P —> Suc + Pi
R11. NDP kinase:	UDP + ATP <—> UTP +ADP
R12. sucrose transporter:	eSuc —> Suc
R13. ATP consumption:	ATP —> ADP + Pi
R14. starch synthesis:	G6P + ATP —> 2Pi + ADP + starch



BIOCHEMISTRY, EXAMPLE2

POTATO TUBER

- monochromatic representation (except R11, R13)
- unambiguity ?
- analysis ?

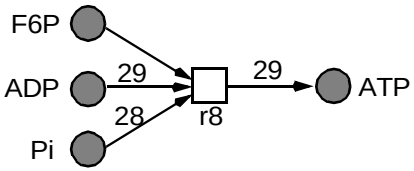


BIOCHEMISTRY, EXAMPLE2

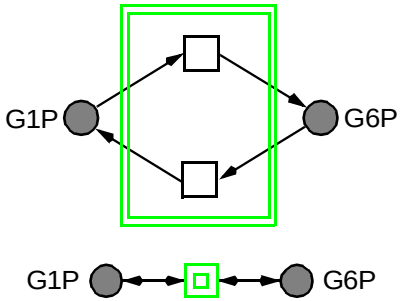
POTATO TUBER

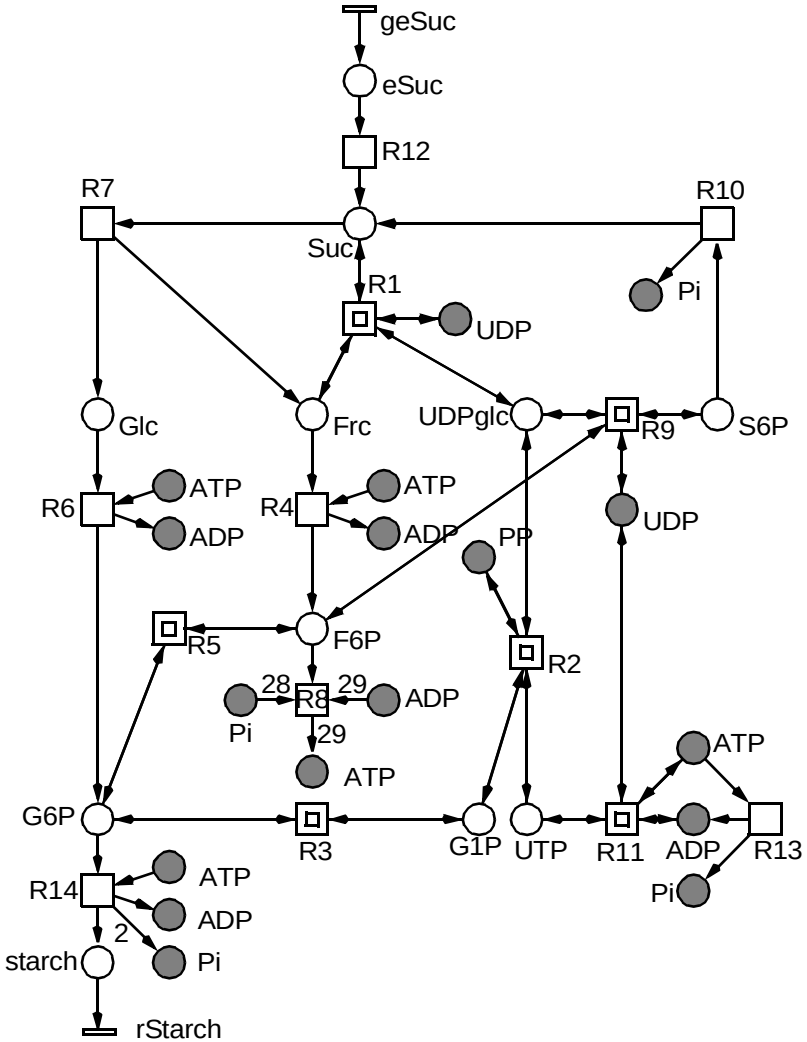
- Petri net
- T-invariants: elementary modes in steady state

R8: F6P + 29 ADP + 28 Pi –> 29 ATP



R3: G1P <–> G6P





BIOCHEMISTRY, SUMMARY

PLACES

- chemical compounds

TRANSITIONS

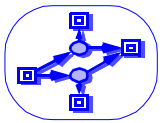
- chemical reactions

FLOW ARCS

- chemical reactions' input & output compounds
- metabolic pathways: stoichiometric relations -> arc weigths

TOKENS

- presence of chemical compounds
- number of available units (molecules)



MILAN CONCEPT, OBJECTIVES

- **analysis of traditional music pieces**

  -> *concise visualization of underlaying musical structures*

- **design and creation of music pieces following a given musical idea**

  -> *computer-based music composition*

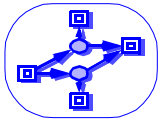
BASIC INGREDIENTS

- **music objects, given in MIDI norm (Musical Instruments Digital Interface)** **-> tokens**

  -> *musical basic material* -> *initial tokens*

- **algorithms to process music objects by traditional composition techniques** **-> transition inscriptions**

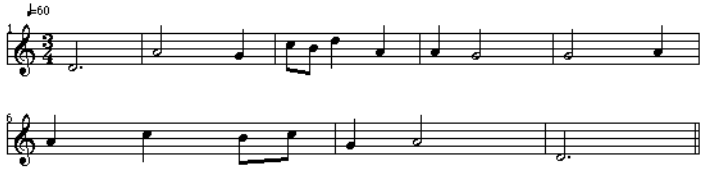
  -> *algorithmic composition theory by Schönberg*



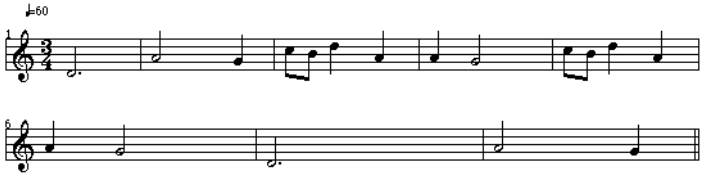
MUSIC, EXAMPLE

THE GIVEN MUSIC TO BE ANALYZED

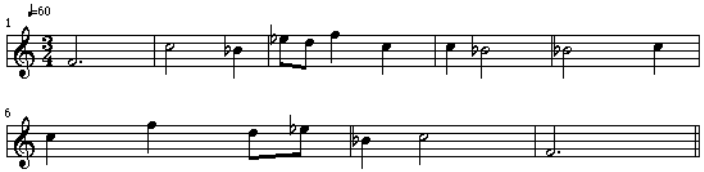
- six melodies
- eight measures each



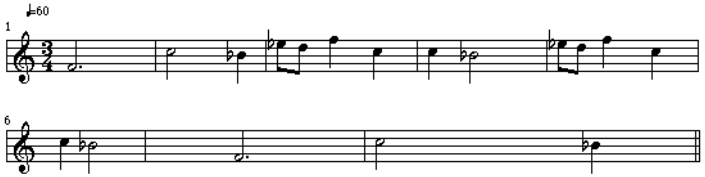
(a) melody, consisting of the phrase A (bar 1-4) and the inversion of A (bar 5-8)



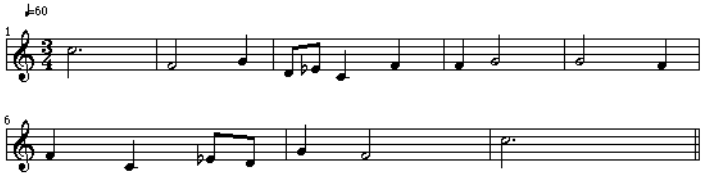
(b) melody, consisting of the phrase A and a rotation of A



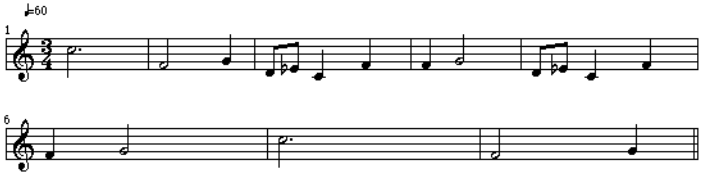
(c) melody, consisting of a transposition T of the phrase A and the inversion of T



(d) melody, consisting of a transposition T of the phrase A and a rotation of T



(e) melody, consisting of a mirage M of the phrase A and the inversion of M



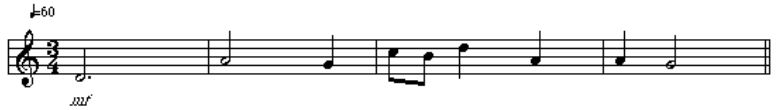
(f) melody, consisting of a mirage M of the phrase A and a rotation of M

MUSIC, EXAMPLE

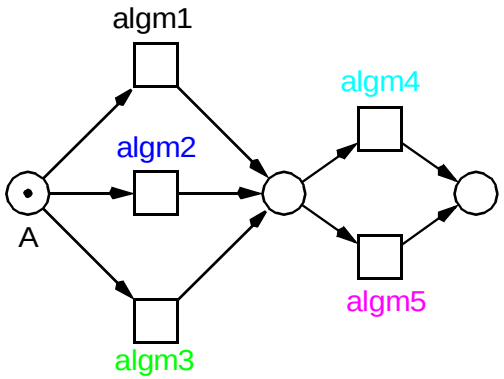
PETRI NET MODEL

- nondeterministic
- six net executions
  - six melodies
- net structure:
  - -> composition principles
  - -> melodies' relationsship

basic material of these melodies - phrase A of four measures length
-> music object in m_0



the predefined music object A is transformed during the net execution according to the five algorithms:



algm1 := keep each note of the music object;

algm2 := transpose each pitch of the music object by 3 semitones upwards;

algm3 := transpose each pitch of the music object regarding g1 as the mirror note;

algm4 := invert all notes of the music object;

algm5 := shift the music object by three notes to the left;

MUSIC, SUMMARY

PLACES

- subsection (bars) of a music piece

TRANSITIONS

- algorithms to transform music objects

FLOW ARCS

- composition principles

TOKENS

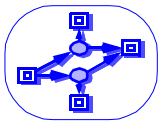
- music objects

CASE STUDIES

- traditional compositions
- Bolero by Ravel
- Sacre du Printemps by Strawinsky

REFERENCES

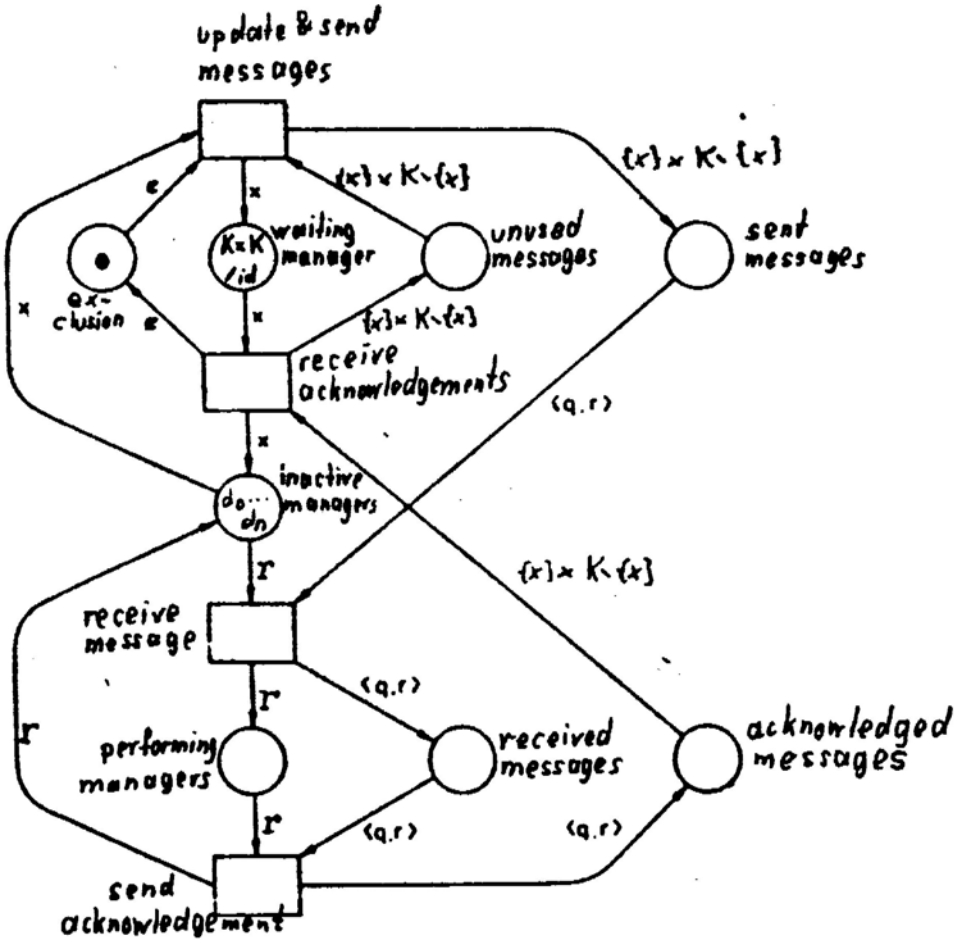
- University Milan, Laboratorio di Informatica Musicale, 198x
- Levens, U. M., 1995, http://www.informatik.uni-oldenburg.de/~levens

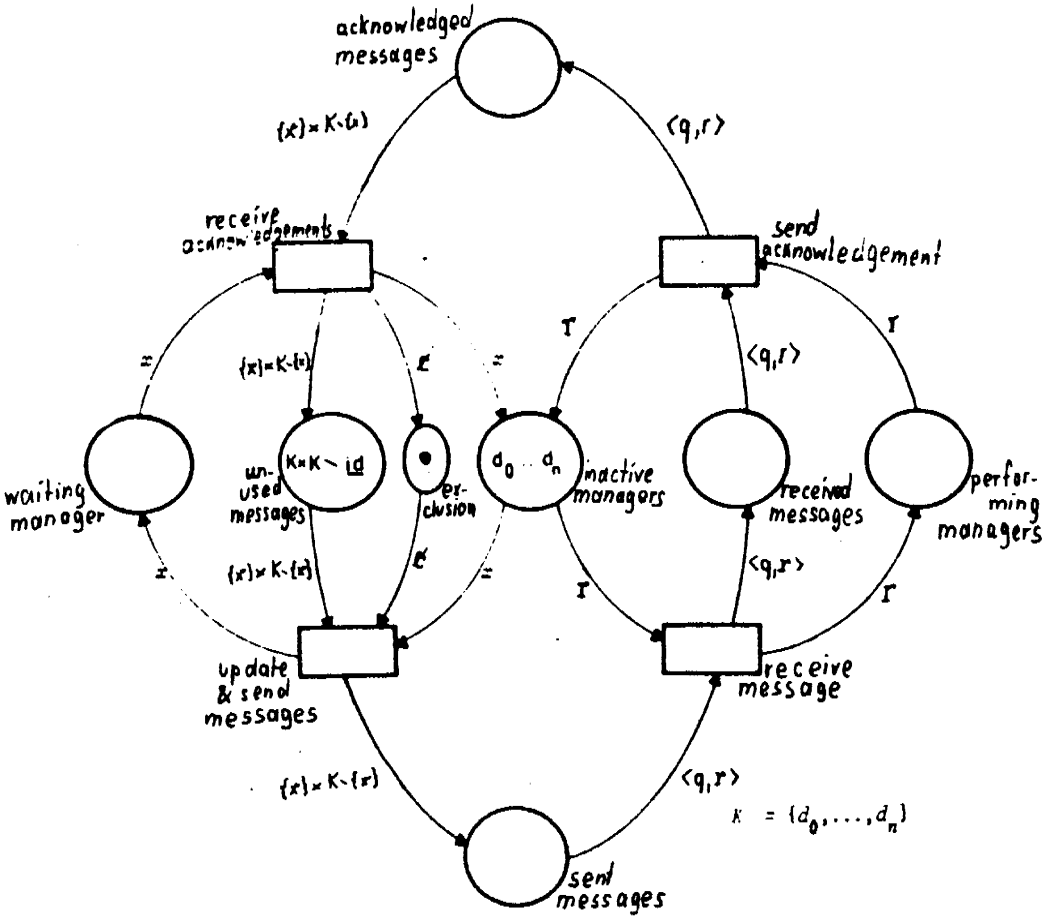


AESTHETICS, EXAMPLE1

RIGHT LAYOUT ?

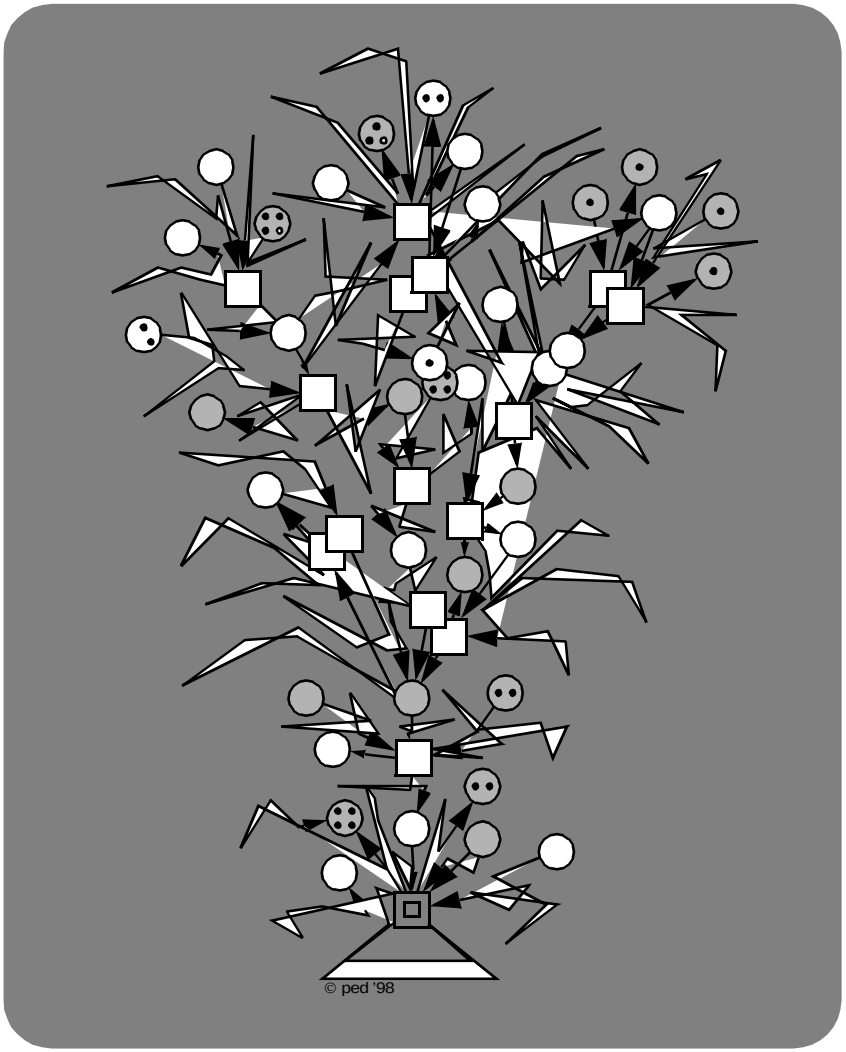
[PN Newsletter 10, February 1982]





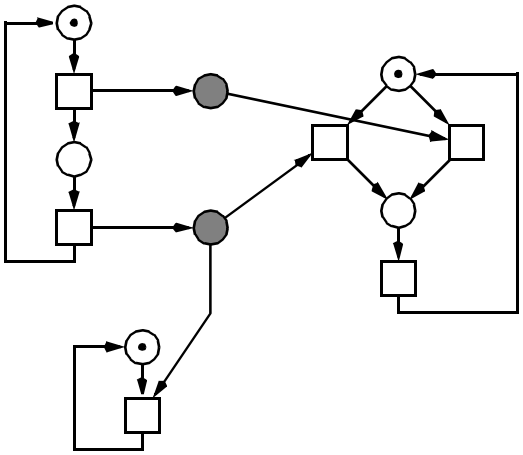
AESTHETICS, EXAMPLE2

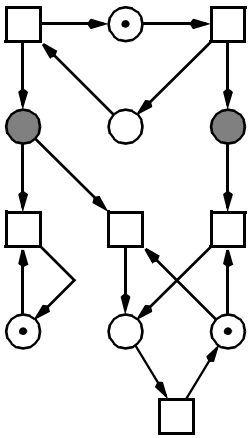
FLOWERS

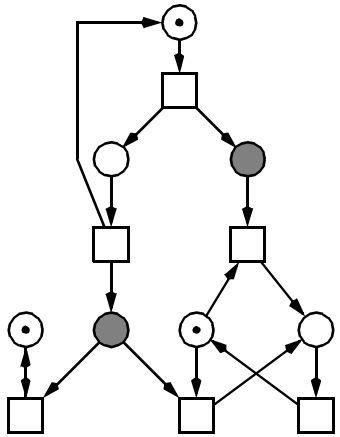


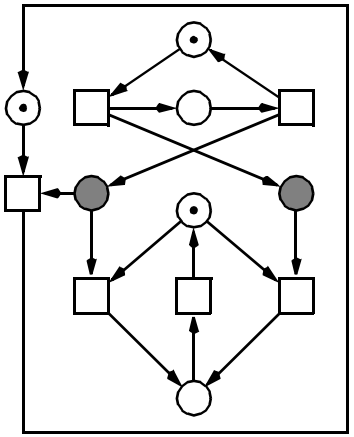
AESTHETICS, EXAMPLE3

WHAT HAVE THESE NETS IN COMMON ?









AESTHETICS, JUGENDSTIL PETRI NETS I

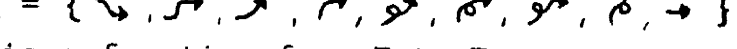
[PN Newsletter 17, June 1984]

Definition

$\mathcal{J} = (S,T;F,F_{jug})$ is called a jugendstilnet iff

(i) $(S,T;F)$ is a net,

(ii) F_{jug} is a finite set of arcs with

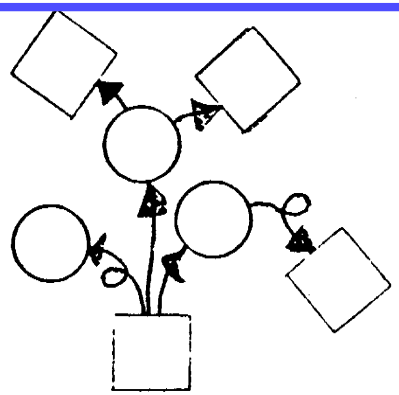
$F_{jug} = \{$  $\}$

(iii) jug is a function from F to F_{jug}.

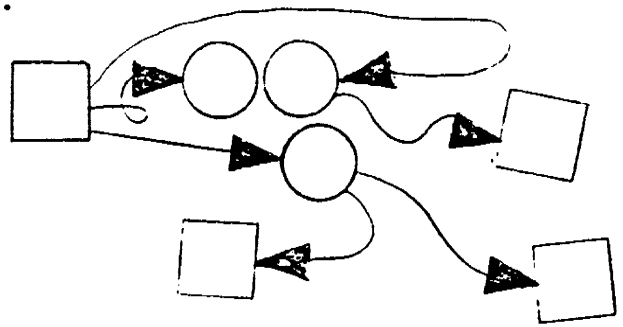
Note, that the length of the arcs is not determined. Graphically we represent $t \in T$ as boxes and $s \in S$ as circles as usual, but instead of denoting $f \in F$ as a boring arc we use jug(f) in its place.

Let's consider some examples:

1.



2.



AESTHETICS, JUGENDSTIL PETRI NETS II

Esthetic construction rule 1

Don't use arcs with loops for inner arcs, i.e.

$\bigwedge_{f \in F}$ (f is an inner arc of x $\Rightarrow$ jug(f) contains no loop)

Esthetic construction rule 2

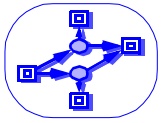
Inner arcs are always to be longer than outer arcs.

Esthetic construction rule 3 (transitive closure of rule 2)

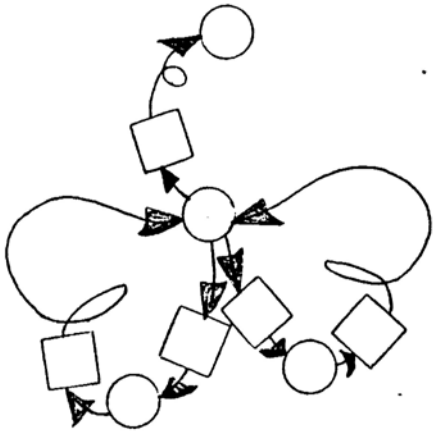
The more "inner" the arc the longer should it be.

Esthetic construction rule 4

Try to give the whole net the form of some geometric figure.



AESTHETICS, JUGENDSTIL PETRI NETS III



Translation method

Step 1:

Look at the definition of that petrinet you want to translate: Surely, you find two disjoint sets S and T and a flowrelation F; all the other things you can forget.

Step 2:

Find a function jug (but don't forget considering the esthetic construction rules 1 - 4 !).

Although this method seems to be very simple, it turns out to be rather complicate especially for untrained users.

Theorem

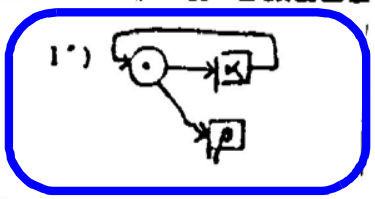
Every kind of petrinet can be translated into a jugendstilnet by using the translation method given above.

!

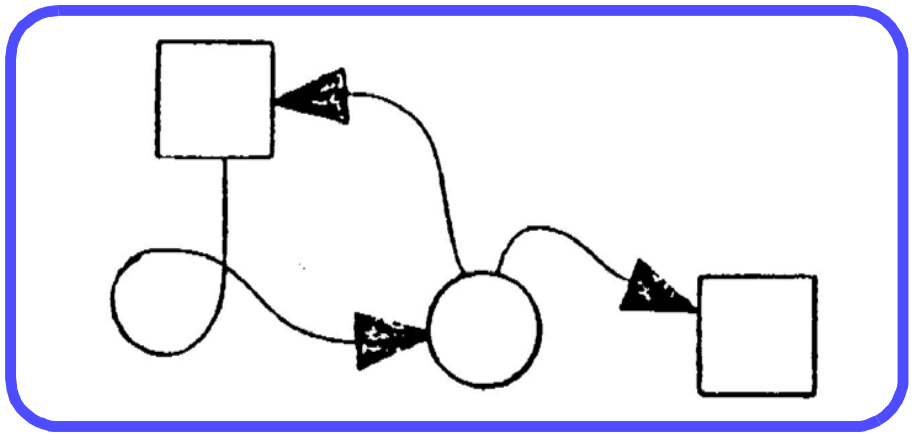
AESTHETICS, JUGENDSTIL PETRI NETS IV

Example 1 (taken from [GM])

We see a small CCS-net.



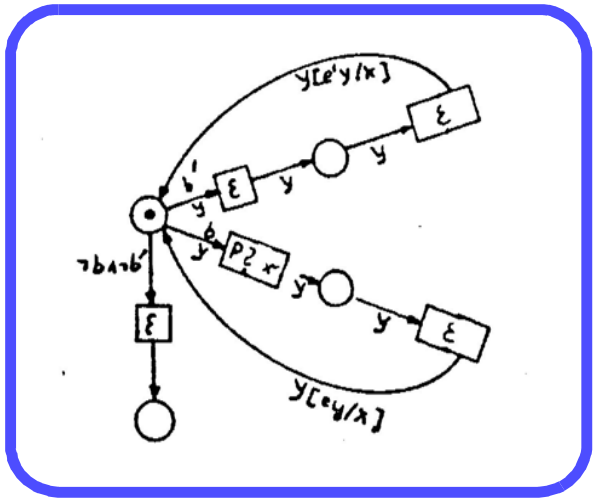
Certainly, the petrinet-insider is deeply impressed, fore the sweat of hard thinking is obvious. Nevertheless, outsiders won't take a further look at it. So, we give a jugendstil translation for it:



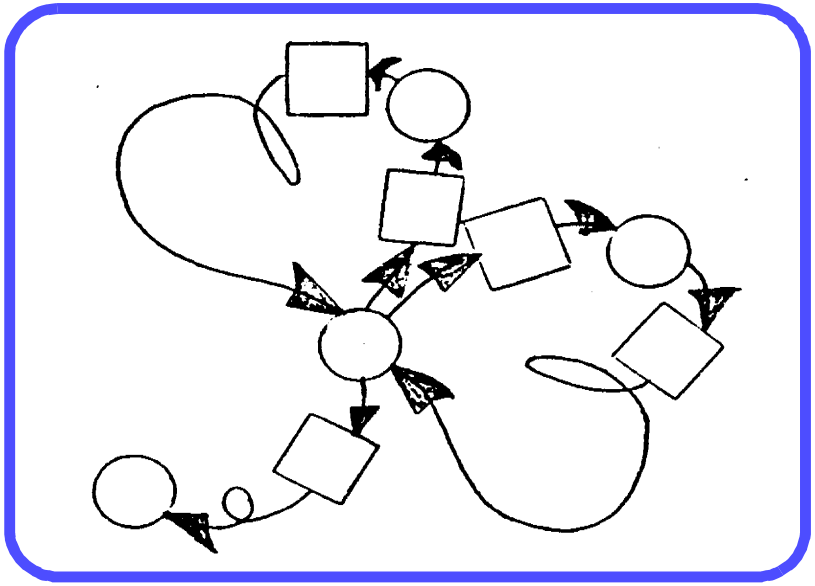
AESTHETICS, JUGENDSTIL PETRI NETS V

Example 2 (takenfrom [GR])

The following net is a CSP-net, where communication hasn't been introduced yet.



What we criticize here is twofold.
Firstly, the outer frame is rather boring, although we don't call it ugly.
Secondly, those arc inscriptions might bring confusion to the viewer, they have to be abolished.
This is our suggestion:



MORE . . . ?

SEASONAL GREETINGS

Merry Christmas &
Happy New Year !

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