Verificarea protocoalelor

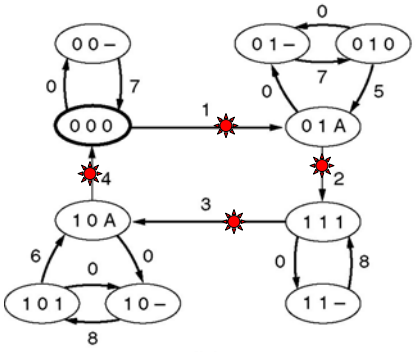
Verificarea Protocoalelor

Specificarea

- ne-formala (limbaj natural)
- formala - modele
  - tranziţionale
    - automate - FSMs (Finite State Machines)
    - reţele Petri
  - algoritmice
  - hibride – FDTs (Formal Description Techniques)
    - Estelle (ISO 9074)
    - LOTOS (**L**anguage **O**f **T**emporal **O**rdering **S**pecification) – ISO 8807
    - SDL (**S**pecification and **D**escription **L**anguage) – ITU-T recomandarea Z.100
  - TTCN-3 - Testing and Test Control Notation Version 3

Model de automat finit

- FSM: Finite State Machine (exemplu pentru protocolul 3).
- Reprezinta starile ca un triplet (T, R, C):
  - Transmitator: (0, 1): corespunde cadrului trimis;
  - Receptor: (0, 1): corespunde cadrului aşteptat;
  - Canal: (0, 1, A, -): corespunde conţinutului canalului.
- Stare iniţială: (0,0,0)

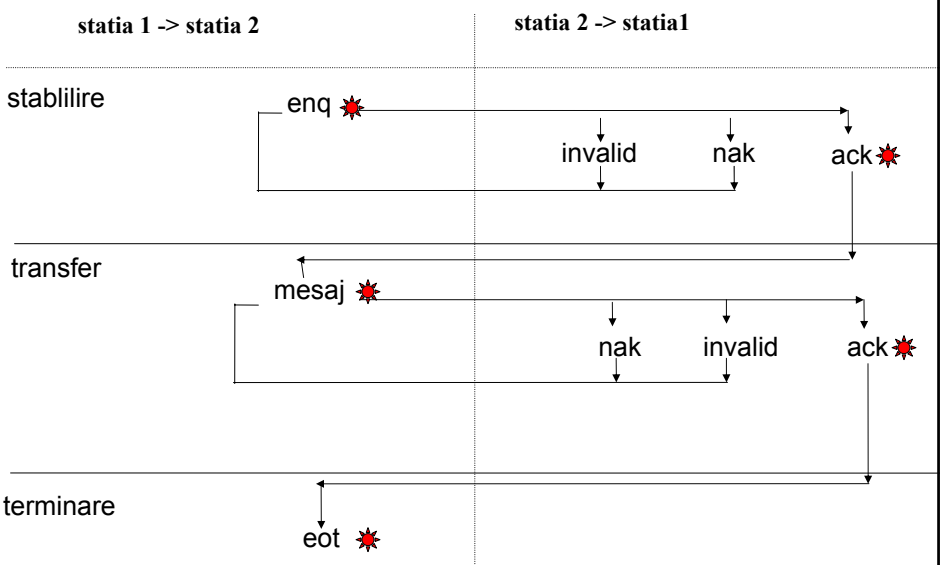


(a)

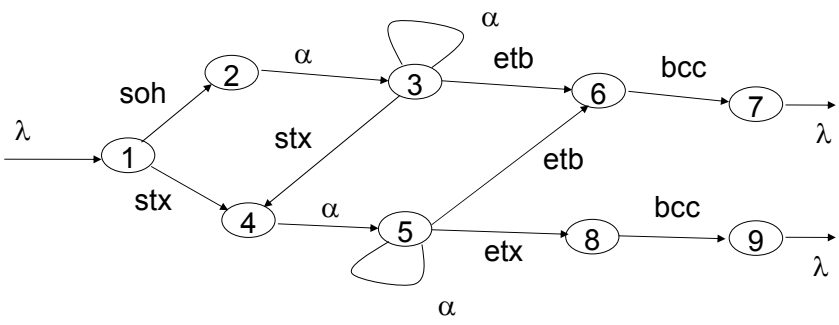
Transition	Who runs?	Frame accepted	Frame emitted	To network layer
0	–	(frame lost)		–
1	R	0	A	Yes
2	S	A	1	–
3	R	1	A	Yes
4	S	A	0	–
5	R	0	A	No
6	R	1	A	No
7	S	(timeout)	0	–
8	S	(timeout)	1	–

(b)

Modelul diagramelor de tranziţii



Modelul transmițătorului / receptorului



Tabel acțiuni

Acțiuni:

a - inițializare tampon mesaj;

b - inițializare tampon text şi variabile text;

c - start text transparent;

d - primul caracter; pune contor pe valoarea 1;

e - alte caractere de text; increment contor;

f - sfârşit antet, memorează şi interpretează început text;

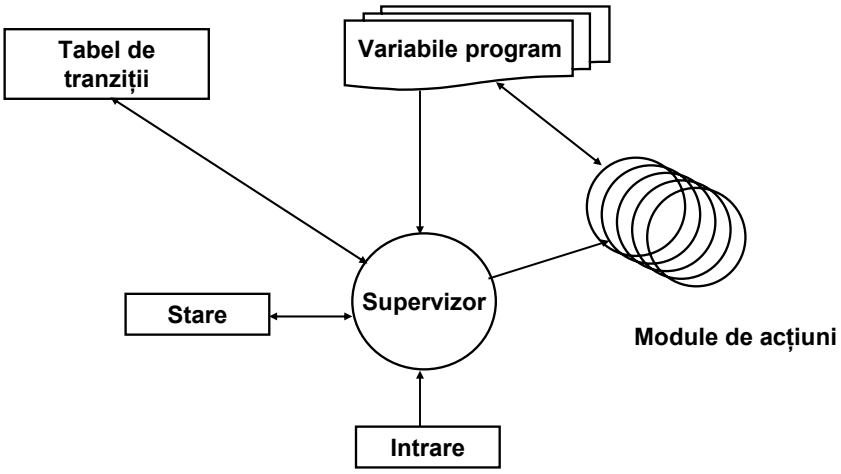
g - verificare erori;

k - sfârşit bloc/text; aşteaptă caracter verificare.

stare\intr	soh	stx	α	dle	syn	etb	etx	β	itb	bcc
1	2 a	4 b		10 c						
2			3 d		2					
3		4 f	3 e		3	6 k				
4			5 d		4					
5			5 e		5	6 k	8 k			
6										7 g
7										
8										9 g
9										

stare următoare / acțiune

Implementare



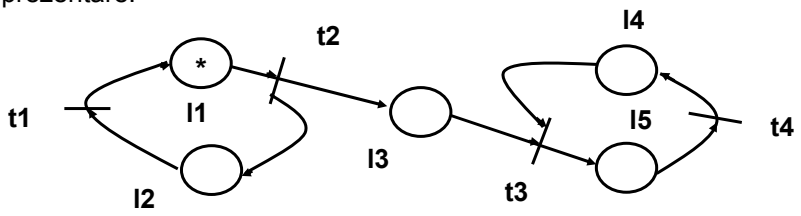
Rețele Petri

Rețele Petri

- Propuse în forma iniţială de Carl Adam Petri în 1962;
- Au evoluat spre mai multe variante:
  - Reţele Petri elementare;
  - Reţele Petri generalizate;
  - Reţele Petri cu arce inhibitoare;
  - Reţele Petri colorate;
  - Reţele Petri continue;
  - Reţele Petri cu predicate;
  - Reţele Petri cu capacităţi;
  - Reţele Petri cu priorităţi.
- Unele variante sunt echivalente între ele, altele nu.
- Se folosesc pentru analiza (automată) a proprietăţilor sistemelor concurente, bazate pe evenimente:
  - Caz particular: verificarea protocoalelor.

Definiții (1)

- Reţea Petri este **RP = (L, T, I, O)**
  - L, mulţime finită de locuri;
  - T, mulţime finită de tranziţii, **L ∩ T = Φ**;
  - I, funcţie de incidenţă înainte **I : L x T → {0, 1}**;
  - O, funcţie de incidenţă înapoi **O: T x L → {0, 1}**.
- Un Marcaj **M : L → N** asociază fiecărui loc un număr natural.
- Pentru fiecare t din T se definesc:
  - Mulţimea locurilor de intrare: **Pre(t) = {l ∈ L | I(l, t)<>0}**;
  - Mulţimea locurilor de ieşire: **Post(t) = {l ∈ L | O(t, l)<>0}**.
- Reprezentare:

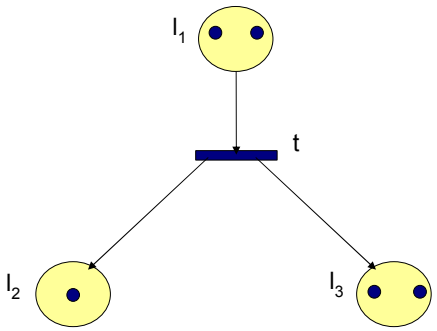


Definiţii (2)

- Tranziţia t este **executabilă** pentru marcajul M dacă:
  - M(l) >= I(l,t), ∀ l ∈ Pre(t) (fiecare loc de intrare are >= un punct).
  - deoarece I(l,t)=0 pentru l ∉ Pre(t) → **M(l) >= I(l,t), ∀ l ∈ L**
- **Execuţia** tranziţiei t in M schimbă marcajul în M':
  - M'(l) = M(l) - I (l,t) + O (t,l), ∀ l ∈ L
- Pentru execuţia unei tranziţii se foloseşte notaţia:
  - M -t-> M'
- O **secvenţă posibilă de execuţii** din M în M', notată M =$=> M', este **$ = $t_1, t_2, t_3, ... t_n$** dacă ∃ marcajele M_1, M_2, M_3, ... M_n a.î. M -t_1-> M_1 -t_2-> M_2 -t_3-> M_3 ... -t_n-> M_n=M'
- Fie RP şi M_0 un marcaj iniţial. Clasa marcajelor accesibile din M0 este notată $\mathcal{A}(M_0)$.

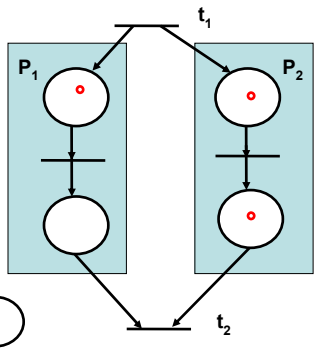
Execuţia unei tranziţii

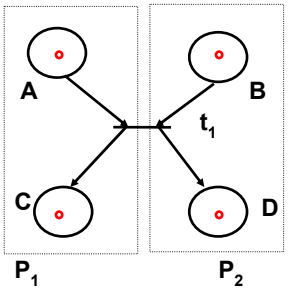
- Fie:
  - Locurile l1, l2, l3;
  - Marcajul iniţial M = [2, 0, 1];
  - Tranziţia t, executabilă.
- Execuţia este **instantanee**.
- Marcajul după execuţia tranziţiei M' = [1, 1, 2].
  - Fiecare loc de intrare pierde un punct;
  - Fiecare loc de ieşire primeşte un punct.

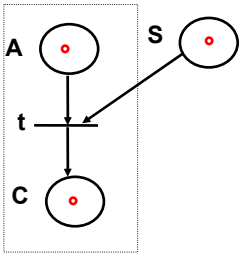


Ce modelăm?

- Paralelismul:
  - P_1 şi P_2 se execută în paralel.
- Sincronizarea:
  - Rendez-vous.
  - Semafor.



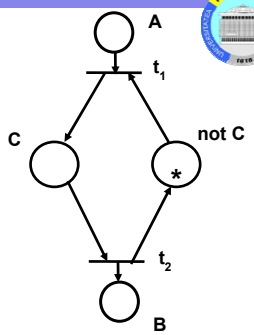


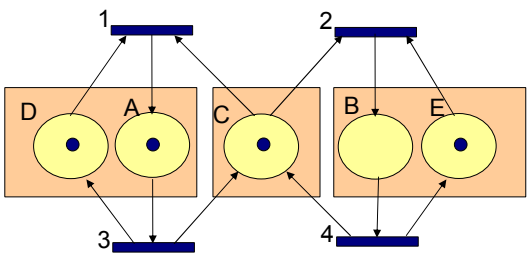


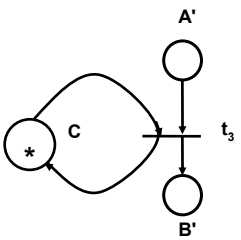
Ce mai modelăm?

- Memorarea unei condiţii şi a opusului său.
- Citirea unei condiţii, fără modificarea ei.
- Excluderea mutuală.

Resursa e eliberată, poate fi refolosită

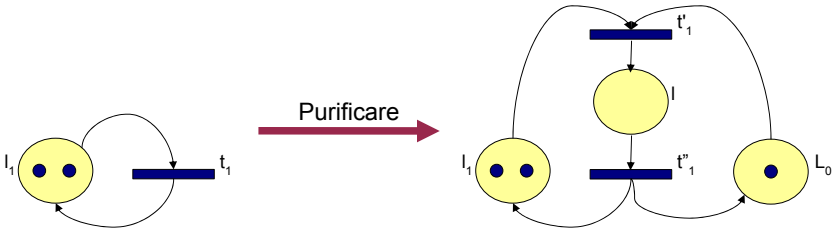






Rețele Petri pure

- O tranziţie **impură** are locuri de intrare care sunt şi de ieşire:
  - Pre(t) ∩ Post(t) <> Φ.
- O rețea Petri e **impură** dacă are cel puțin o tranziție impură.
- Rețelele impure mai dificil de analizat → **purificare**.
- Algoritm de purificare (creşte **dimensiunea** rețelei!):
  - Se adaugă **un loc special** L_0, cu marcaj 1;
  - Se transformă **fiecare tranziție** t într-un ansamblu de 2 tranziții t', t" şi un loc intermediar l, cu marcaj 0;
  - Se adaugă un arc de la L_0 la fiecare tranziție t' şi un arc de la fiecare tranziție t" către L_0.
- Exemplu:



Modelarea protocolului start-stop

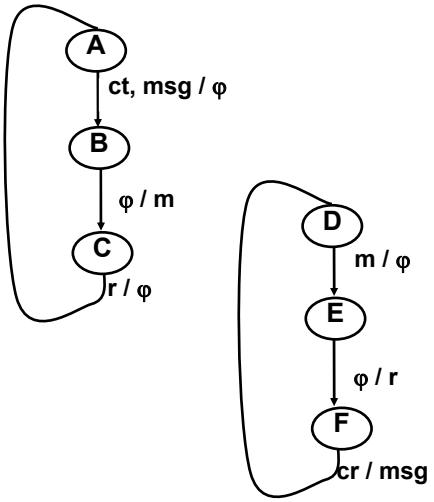
- Model algoritmic:
  - Transmiţător:

```
do
{
  asteapta cerere emisie (ct,msg)        A
  pregateste mesaj (msg,m)
  transmite mesaj (m)                    B
  asteapta confirmare (r)                C
} forever;
```

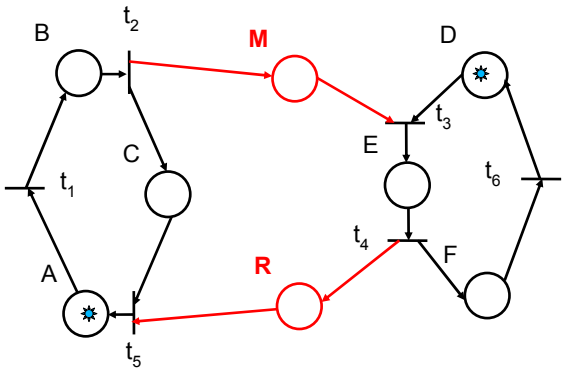
  - Receptor:

```
do
{
  asteapta mesaj (m)                     D
  pregateste raspuns (m,r, msg)
  transmite confirmarea (r)              E
  asteapta cerere receptie (cr)
  transfera mesaj (msg)                  F
} forever
```

- Model de automat finit:



Rețeaua Petri asociată



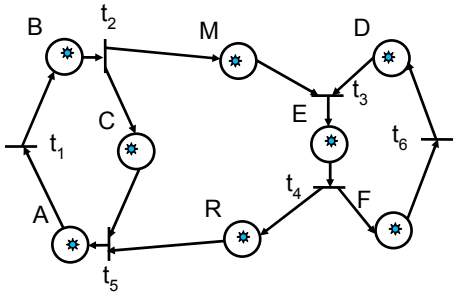
Tranziții:

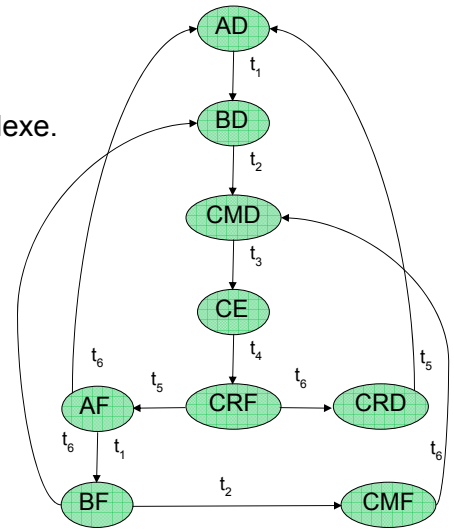
t1 preluare mesaj produs de utilizatorul transmițător;
t2 transmitere mesaj mediului de comunicare;
t3 recepție mesaj de la mediu;
t4 transmitere confirmare;
t5 recepție confirmare;
t6 consumare mesaj de utilizator receptor.

Marcajul corespunzător unei stări inițiale M0:
Entitatea emițătoare aşteaptă producerea unui mesaj (A);
Entitatea receptoare este pregatită pentru recepție (D);
Mediul de transmisie este gol.

Maşina de puncte

- Diagramă de tranziţii.
- Explorează marcajele posibile.
- Dificil de folosit la sisteme complexe.



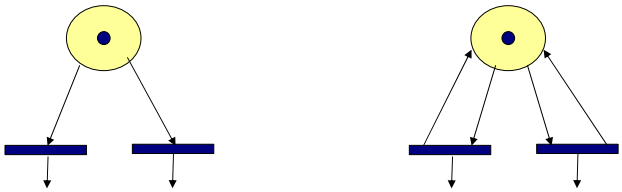


Validarea protocoalelor - proprietăți generale

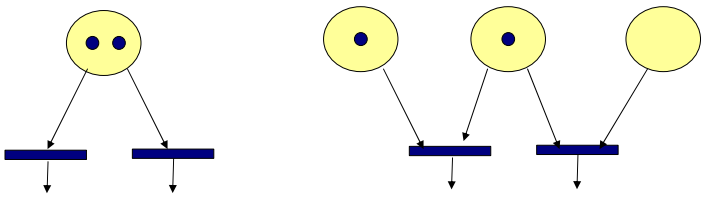
- Mărginire:
  - Orice M ϵ $\mathcal{A}$(M_0) şi orice l din L ➔ M(l)<=n.
- Siguranţă:
  - Mărginire pentru n=1.
- Viabilitate:
  - Din oricare M accesibil din M_0 există o secvenţă de execuţii care conţine t.
- Cvasi-viabilitate:
  - există o secvenţă de execuţii din M_0 care conţine t.
- Home state:
  - Din oricare M accesibil din M_0 există o secvenţă de execuţii care conduce în H.

Persistența

- Conflicte efective:



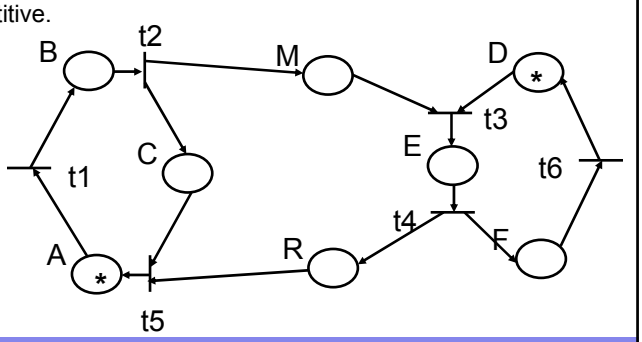
- Conflicte structurale, dar ne-efective:



Persistență: În orice M accesibil din M0, în care tj şi tk sunt executabile, tj, tk şi, prin simetrie tk, tj sunt secvențe posibile de execuții din M.

Validarea protocoalelor - proprietăți specifice

- Invarianţi:
  - Pe locuri (L-invarianţi):
    - M(A) + M(B) + M(C) = 1, pentru orice M ∈ $\mathcal{A}(M_0)$.
    - Componentă / reţea conservativă.
  - Pe tranziţii (T-invarianţi):
    - Avans sincron 0 <= N(t3) - N(t4) <= 1.
    - Secvenţe repetitive.

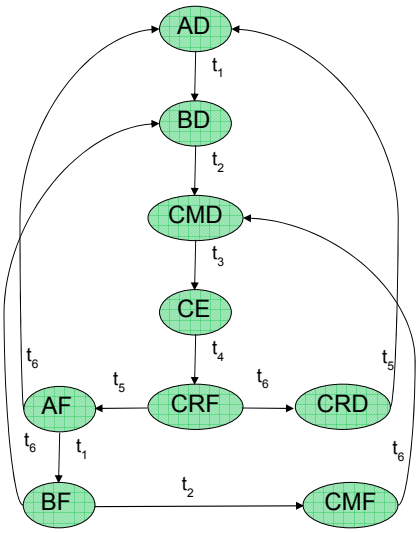


Validarea protocoalelor - metode

- Maşina de puncte.
- Arbori şi grafuri de acoperire.
- Calculul invarianţilor (model algebric).
- Reducerea modelelor.

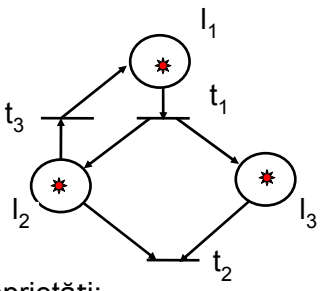
Maşina de puncte

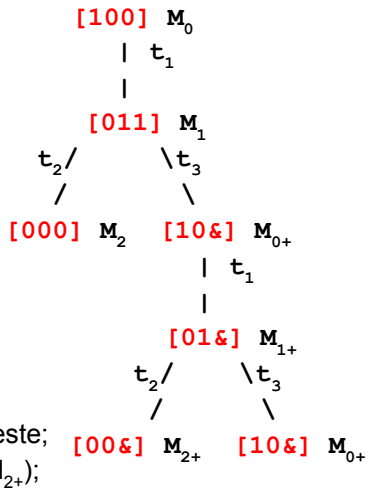
- Proprietăţi:
  - sigură;
  - viabilă;
  - home state;
  - **invarianţi?**



Arbori şi grafuri de acoperire

- Analiza reţelelor nemarginite.





- Proprietăţi:
  - Locurile l_1 şi l_2 sunt mărginite; l_3 nu este;
  - Există o infinitate de blocări (M_2 si M_{2+});
  - RP este cvasi-viabilă.

Construire arbore de acoperire

```
construire_arbore_acoperire()
{
  calculeaza succesoarele lui M0;
  for (fiecare succesor M)
    if (M>M0)
      marcheaza cu & fiecare componenta a lui M superioara componentei
      corespunzatoare din M0;
    while (exista un marcaj nou Mi, neconsiderat)
      if (nu exista pe calea de la M0 la Mi un marcaj Mj=Mi)
      {
        calculeaza succesoarele lui Mi;
        for (fiecare succesor Mk al lui Mi)
        {
          o componenta & a lui Mi ramine & in Mk;
          if (exista un marcaj Mj pe calea de la M0 la Mk cu Mk>Mj)
         marcheaza cu & fiecare componenta din Mk superioara
             componentei coresp. din Mj;
        }
      }
}
```


Analiza RP prin calculul invarianților

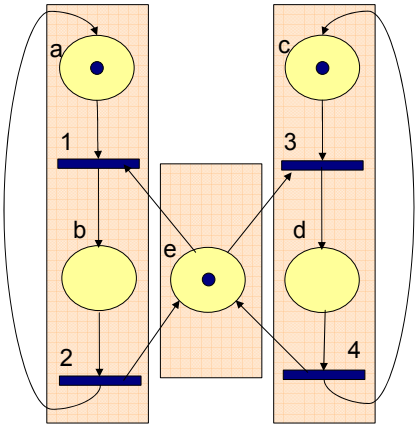
- Fie RP o reţea Petri **pură** (fără bucle), în care L şi T sunt ordonate (arbitrar):
  - L: l1 < l2 < ...< lm,
  - T: t1 < t2 < ...< tn.
- Matricea A : L x T → Z cu A [li, tj] = O (tj, li) – I (li, tj) este **matricea de incidenţe** a lui RP.
- Notăm:
  - A [li, -] = linia li;
  - A [-, tj] = coloana tj.
- **L-vector** = o matrice coloană indexată după L.
- **T-vector** = o matrice coloană indexată după T.

Modelul excluderii mutuale

Matricea de incidenţe:

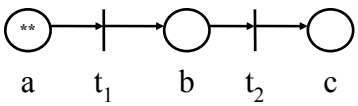
A	1	2	3	4
a	-1	1		
b	1	-1		
c			-1	1
d			1	-1
e	-1	1	-1	1

A[a,1] = O[a,1] - I[a,1] = 0-1 = -1

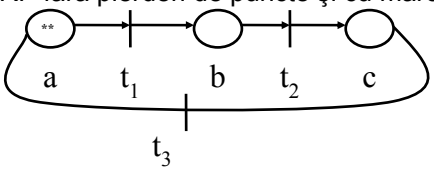


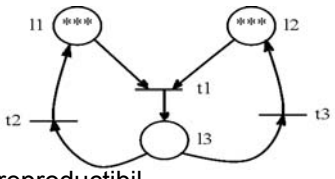
Aspecte de corectitudine

- Aspecte de corectitudine:
  - Garantare că nu se pierd puncte;
  - Posibilitate reproducere marcaje.



- Exemple:
  - RP fără pierderi de puncte dar cu marcaj nereproductibil.
  - RP fără pierderi de puncte şi cu marcaj reproductibil;





  - RP cu pierderi de puncte şi cu marcaj nereproductibil.

L-invarianți

- Daca M şi M' au M –t-> M'; rezulta
  - M' = M + A[-, t];
- Pentru modelul excluderii mutuale avem **invariantul** :

  `M[a]+2M[b]+M[c]+2M[d]+M[e] = 3` (orice M).

- Pentru g^T **= [1, 2, 1, 2, 1]** şi M, M' reprezentaţi ca **L-vectori**
  - $g^T.M = g^T.M' = g^T.M + g^T.A[-,t]$.
- Rezultă:
  - $g^T.A[-,t] = 0$;
  - $g^T.A = 0$ (relaţia anterioară valabilă pentru orice t).
- **g este L-invariant**.

- Un L-vector I este un L-invariant ⇔ $I^T.A = 0$.
- Un L-invariant ne-negativ I se numeşte **minimal** ⇔ nu există un I' a.i. 0 < I' < I.

Exemplul excluderii mutuale (1)

U|A						1	2	3	4
a	1	0	0	0	0	-1	1		
b	0	1	0	0	0	1	-1		
c	0	0	1	0	0			-1	1
d	0	0	0	1	0			1	-1
e	0	0	0	0	1	-1	1	-1	1
a+b	1	1	0	0	0	0	0	0	0
b+e	0	1	0	0	1	0	0	-1	1

Pentru j=1 se **adaugă** liniile **a+b** şi **b+e**

Exemplul excluderii mutuale (2)

U|A					1	2	3	4
c	0	0	1	0	0		-1	1
d	0	0	0	1	0		1	-1
a+b	1	1	0	0	0			
b+e	0	1	0	0	1		-1	1
c+d	0	0	1	1	0			
d+b+e	0	1	0	1	1			

Pentru j=3 se **adaugă** liniile **c+d** şi **d+b+e**

Exemplul excluderii mutuale (3)

U|A					1	2	3	4
d+b+e	0	1	0	1	1	0	0	0	0
c+d	0	0	1	1	0	0	0	0	0
a+b	1	1	0	0	0	0	0	0	0

$$I1 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 1 \end{bmatrix} \quad I2 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} \quad I3 = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

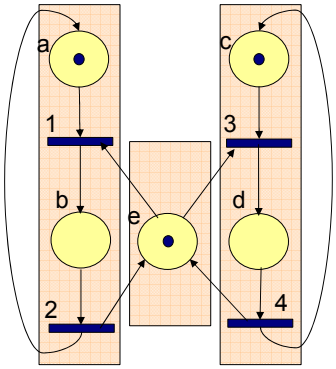
Calcul invarianți

```
calcul_invarianti()
{
  construieste matricea (U|A);
  for (fiecare indice j al tranzitiei tj)
  {
    adauga la matricea (U|A) atâtea linii i câte
      combinatii lineare de câte două linii cu
      coeficienti intregi pozitivi in care se anulează
      elementul [i,j] există;
    elimină din matricea (U|A) liniile i în care
      elementul [i,j] este nenul.
  }
}
```


Folosire invarianți

- Regulă:
  - Dacă M este un marcaj şi I un L-invariant atunci pentru orice M' accesibil din M:
    - **$I^T.M' = I^T.M$**
- Utilizare:
  - Verificarea evitării anumitor marcaje:
    - Dacă există un invariant I a.î. $I^T.M' <> I^T.M$ atunci M' nu poate fi accesibil din M.
  - Găsirea condiţiilor necesare completării unui marcaj M' accesibil din M şi cunoscut parţial;
  - Deducerea unor proprietăţi generale ale marcajelor accesibile.

Exemplu pentru excluderea mutuală



- Invarianţii găsiţi sunt (se omite T=transpus):
  - I_1 = [0 1 0 1 1];
  - I_2 = [0 0 1 1 0];
  - I_3 = [1 1 0 0 0];
  - $I = I_1 + I_2 + I_3$ = [1 2 1 2 1] (invariantul **global**).
- Din $I^T.M = I^T.M0$ obţinem ptr invarianţi:
  - M[b] + M[d] + M[e] =1;
  - M[c] + M[d] = 1;
  - M[a] + M[b] = 1.
- Relaţiile exprimă:
  - Condiţia de excludere mutuală (prima relaţie);
  - Siguranţa: $M[l_i] <= 1$ pentru orice l_i;
  - Reţea conservativă: din $g = I_1 + I_2 + I_3$ se obţine:

    M[a] + 2M[b] + M[c] + 2M[d] + M[e] = 3.

Reproducerea marcajelor

- Efectul tranziţiei 1, scris $M_0 + A[-,1] = M_1$ este echivalent cu:

$$M0 + A.\begin{bmatrix}1\\0\\0\\0\end{bmatrix} = M1$$

- Efectul cumulat al tranziţiilor 1 şi 2 poate fi scris:

$$M0 + A.\begin{bmatrix}1\\1\\0\\0\end{bmatrix} = M0$$

T-invarianți

- T-vectorul J care reprezintă **numărul de execuţii** ale tranziţiilor şi este o soluţie a ecuaţiei A.y = 0 este **T-invariant**:

$$J = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

- J este un T-invariant ⇔ A.J = 0.
- Un T-invariant ne-negativ J se numeşte minimal ⇔ nu există J' a.î. 0 < J' < J.
- Dacă J este un T-invariant atunci există un marcaj reproductibil prin execuţia tranziţiilor în conformitate cu J.
- Pentru modelul excluderii mutuale, RP revine în marcajul iniţial prin execuţia tranziţiilor 1 şi 2 (J1) sau 3 şi 4 (J2).

Calculul T-invarianților

- Din $x^T.A=0$ şi A.y=0 (sau $y^T.A^T$ =0) rezultă că:
  - **T-invarianţii asociaţi lui A sunt L-invarianţii lui A^T.**
- A^T este matricea de incidenţe corespunzătoare RP duale.
- RP dală se obţine astfel:
  - Fiecărui loc în RP îi corespunde o tranziţie în RP dală;
  - Fiecărei tranziţii în RP îi corespunde un loc în RP dală;
  - Fiecărui arc în RP îi corespunde un arc orientat în sens contrar în RP dală.

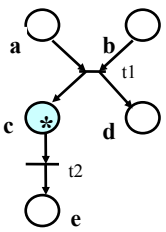
Reducerea RP

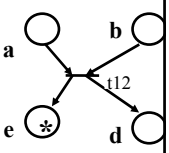
- Reducerea are ca scop scăderea dimensiunilor reţelei.
- Reducerea trebuie să păstreze (cat mai multe) din proprietăţile reţelei.
- Reducere cu păstrarea proprietaţilor generale (marginire, viabilitate,…):
  - R1 (Reducerea unui **loc**);
  - R2 (Reducerea unui **loc implicit**);
  - R3 (Reducerea unei **tranziţii neutre**);
  - R4 (Reducerea **tranziţiilor identice**).
- Reducere cu păstrarea invarianţilor:
  - Ra (Reducerea unei **tranziţii impure**);
  - Rb (Reducerea unei **tranziţii pure**).

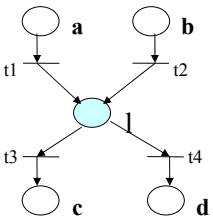
R1: Reducerea unui loc

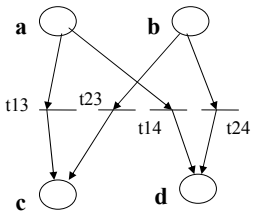
Eliminarea unui loc l:

- Dacă locul l are j intrari şi k ieşiri, ele sunt înlocuite prin j*k tranziţii, obţinute prin contopirea unei tranziţii de intrare cu una de ieşire;
- Ieşirile unei tranziţii de intrare (ex. d - ieşirea lui t1) devin ieşiri ale tranziţiei obţinută prin contopire (t12).





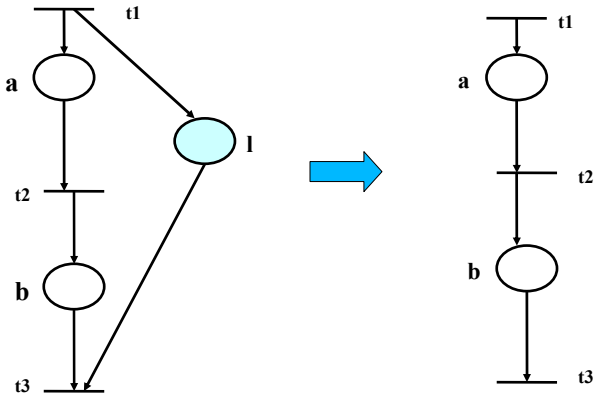




- Daca l este marcat si are k iesiri (locul c din fig), prin eliminarea sa se obtin k retele distincte, marcajul fiind plasat in fiecare caz in locurile corespunzatoare unei alte tranzitii de iesire

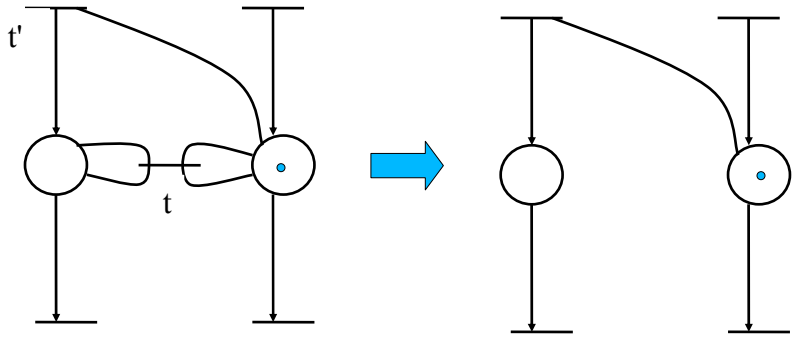
R2: Reducerea unui loc implicit

- Loc implicit:
  - Marcajul său permite întotdeauna execuţia oricărei tranziţii de ieşire, care ar fi executabilă dacă se ignoră l;
  - Marcajul său poate fi determinat din marcajul celorlalte locuri.



R3: Reducerea unei tranziţii neutre

- Tranziţia t este neutră ⇔ Pre(t) = Post(t).
- Eliminare ⇔ există t'<>t cu O(t',l)>=I(l,t) pentru orice l din Pre(t).



R4: Reducerea tranziţii identice

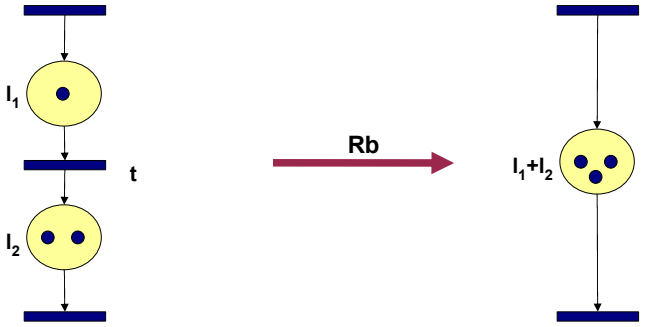
- Tranziţii identice: au aceleaşi locuri de intrare şi de ieşire.
- Dacă există n tranziţii identice, se elimină n-1 din ele.

Ra: Reducerea unei tranziţii impure

- O tranziţie **impură** are locuri de intrare care sunt şi de ieşire:
  - Pre(t) ∩ Post(t) <> Φ.

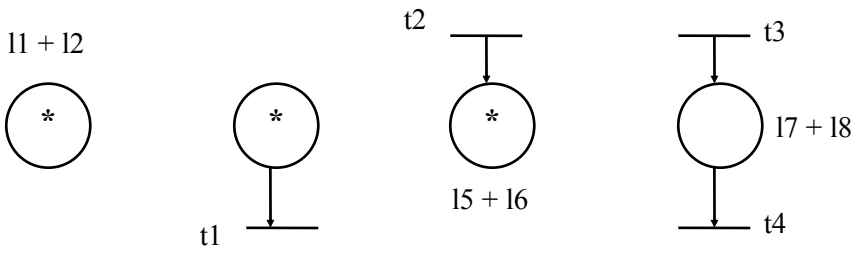
Rb: Reducerea unei tranziții pure

- Reducere tranziţie pură:
  - Se elimină tranziţia pură t;
  - Fiecărui cuplu de locuri l_i din Pre(t) şi l_j din Post(t) i se asociază un loc l_i+l_j al cărui marcaj este $M(l_i)+M(l_j)$.



Cazuri ireductibile

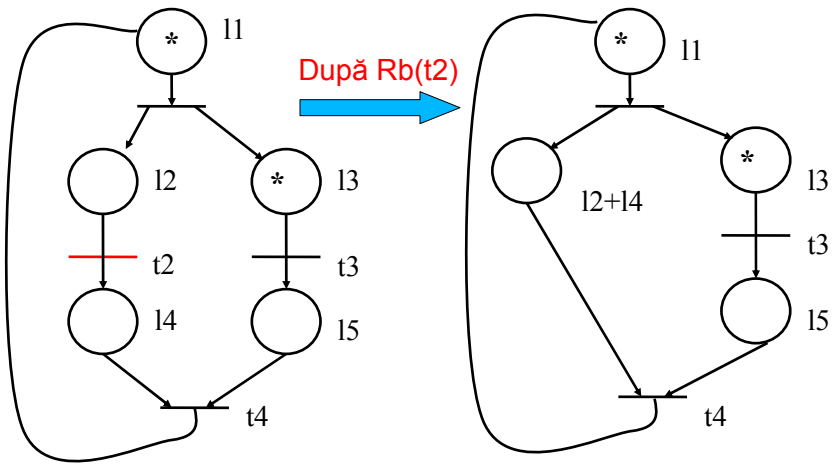
- Reţea conservativă: M(l1)+M(l2)=1.



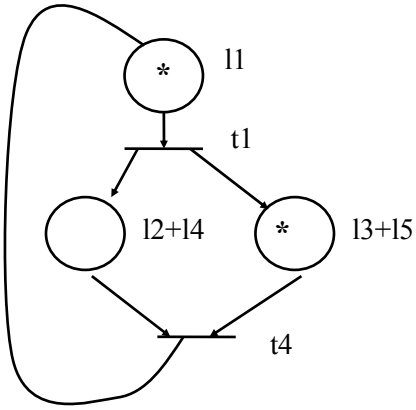
Proprietăți păstrate prin reducere

Reduceri / **Proprietăți**	R1	R2	R3	R4	Ra	Rb
Mărginirea	X	X	X	X		
Siguranța	X		X	X		
Viabilitatea	X	X	X			
Cvasi-viabilitatea	X	X	X	X		
Evitarea blocării	X	X	X	X		
Starea de revenire	X	X	X	X		
Conservabilitatea	X	X	X	X		
Invarianți					X	X

Exemplu reduceri (1)

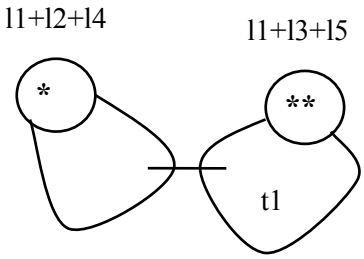


Exemplu reduceri (2)



După Rb(t3)

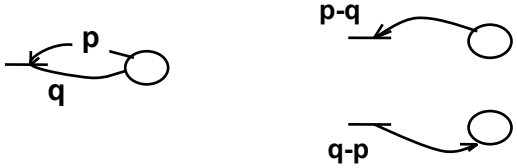
După Rb(t4)



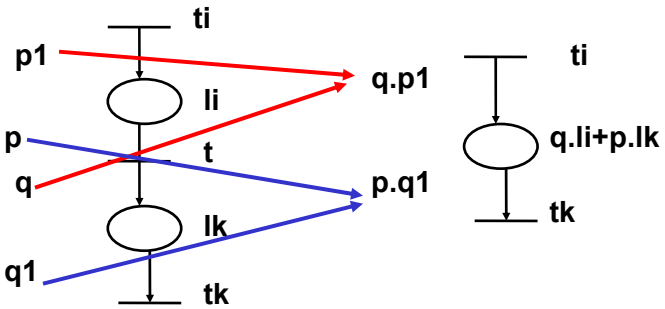
După Ra(t1)

Reduceri pentru RP generalizate

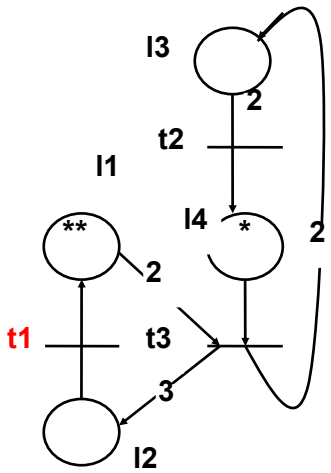
- Reducerea R'a (tranziţie impură):

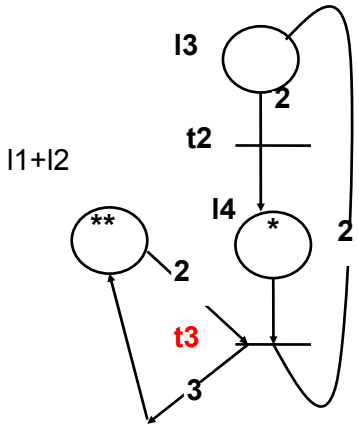


- Reducerea R'b (tranziţie pură):



Exemplu reduceri RP generalizate (1)

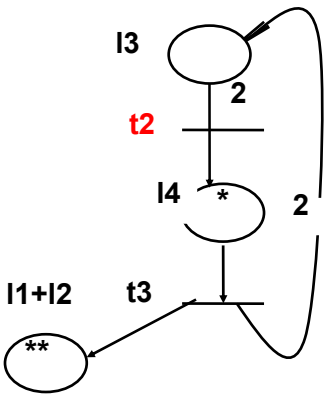




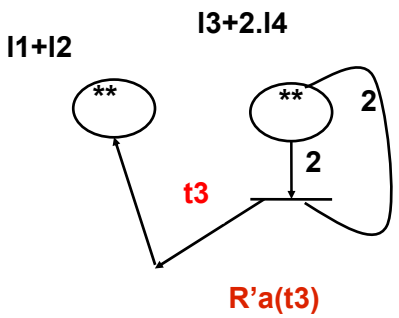
R'a(t3)

Se aplică R'b(t1): M(l1+l2) = 1*0+1*2

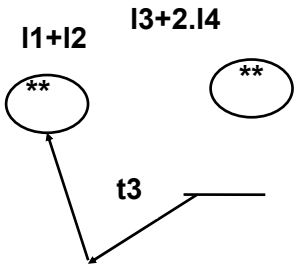
Exemplu reduceri RP generalizate (2)



R'b(t2)



R'a(t3)



Protocol cu bit alternat