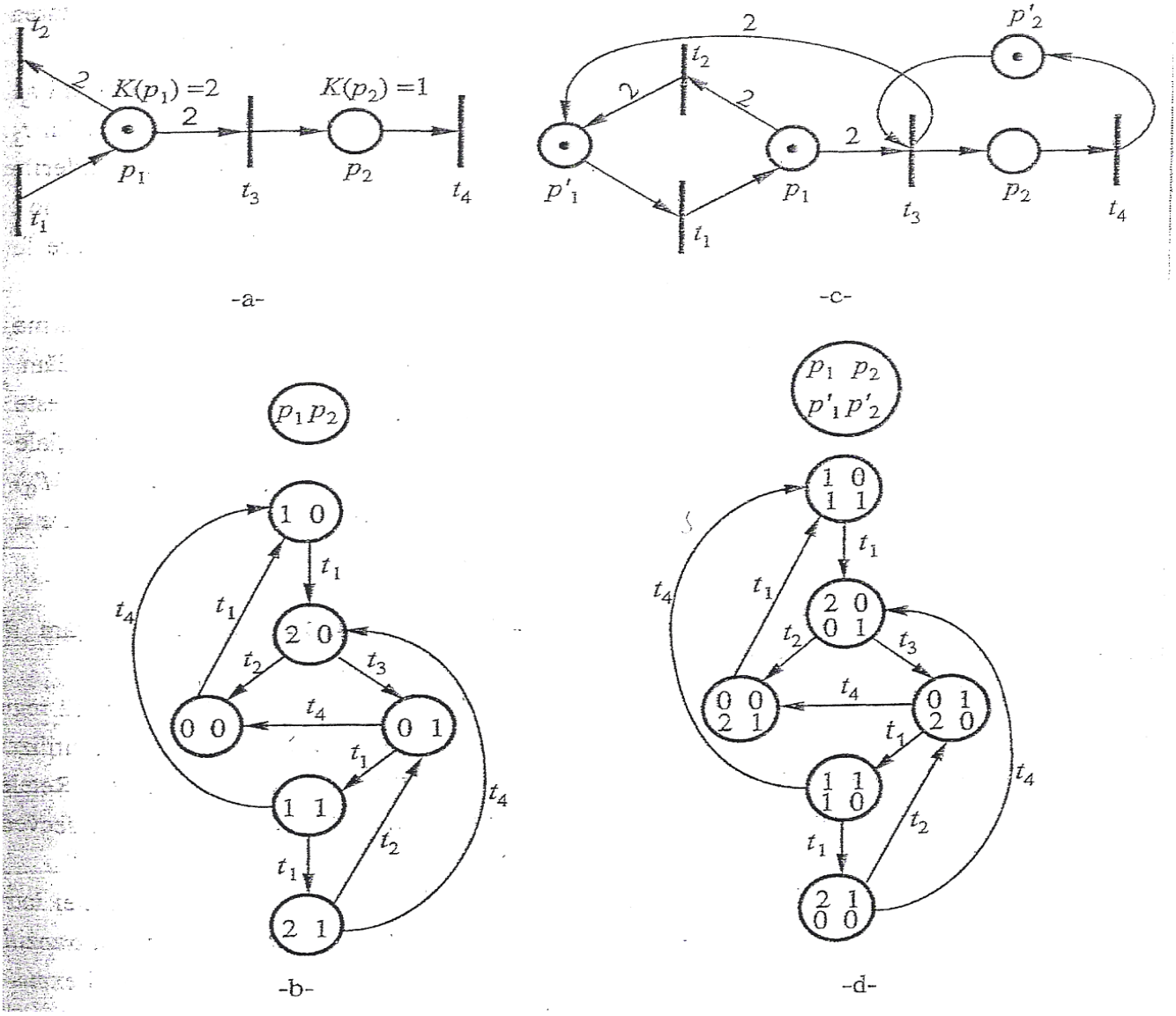


Fig. 2.10 Rețele Petri și grafuri de accesibilitate pentru exemplu
a)Rețeaua(N, M_0) cu capacitate finita;

b)Graful de accesibilitate al rețelei (N, M_0), obținut prin aplicarea regulii stricte a tranziției;

c) Rețeaua(N', M'_0) cu capacitate infinita;

d)Graful de accesibilitate al rețelei (N', M'_0) obținut prin aplicarea regulii simple a tranziției.

Modul de construire a reţelei transformate (*N', M'0*) din fig. 2.10 (c), rezultată din reţeaua iniţială (*N, M0*), prin **aplicarea metodei poziţiilor complementare**:
Pas1. Se adaugă poziţiile complementare p'_1 şi p'_2, precum şi marcajul iniţial aferent acestor noi poziţii şi anume:

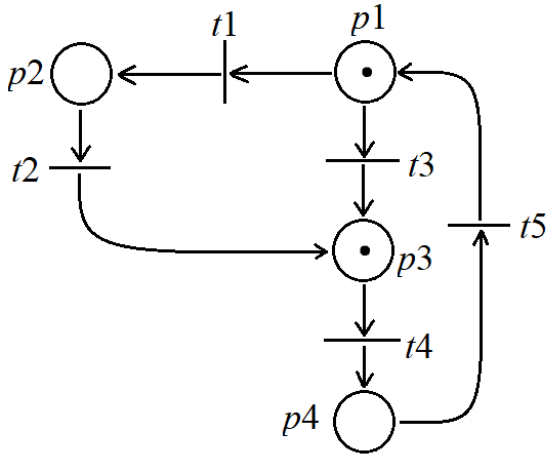
$$M'_0(p'_1) = K(p_1) - M_0(p_1) = 2 - 1 = 1$$
$$M'_0(p'_2) = K(p_2) - M_0(p_2) = 1 - 0 = 1$$

Pas2. Se adaugă arce noi între fiecare tranziţie t şi *unele poziţii complementare*, astfel încât să se menţină aceeaşi sumă de jetoane (şi anume $i = 1, 2$) în fiecare pereche de poziţii complementare p'_1 şi p'_2, atât înainte, cât şi dupa executarea tranziţiei t.

De exemplu, în cazul lui t_1, deoarece $w(t_1, p_1) = 1$, rezultă $w(p_1, t_1) = 1$. În cazul lui t_3, avem $w(t_3, p_1) = w(p_1, t_3) = 2$ şi $w(p'_2, t_3) = w(t_3, p_2) = 1$, întrucât executarea lui t_3 îndepărtează două jetoane din p_1 şi adaugă un jeton în poziţia p_2 (a se remarca ponderile diferite ale celor două arce!). Transformarea se încheie după ce se adaugă şi arcele noi. Corespunzătoare tranziţiilor neluate încă în discuţie (adică t_2 şi t_4), operaţii pe care le recomandăm drept exerciţiu.

Exemplu:

Transformați rețeaua Petri cu capacitate finită din fig. 2.11 într-o rețea Petri cu capacitate infinită.



$K(p_3) = 1$
$K(p_1) = K(p_2) = K(p_4) = \infty$

Fig. 2.11 Rețeaua Petri cu capacitate finită.

Soluție

Utilizând metoda pozițiilor complementare cu referire la poziția p_3, se adaugă poziția p'_3 astfel încât să fie satisfăcute următoarele relații:

$w(t_3, p_3) = w(p'_3, t_3) = 1,$
$w(t_2, p_3) = w(p'_3, t_2) = 1,$
$w(p_3, t_4) = w(t_4, p'_3) = 1,$
$M_0(p'_3) = K(p_3) - M_0(p_3) = 1 - 1 = 0.$

Rețeaua Petri cu capacitate finită din fig. 2.11 este echivalentă cu rețeaua Petri de capacitate infinită prezentată în fig. 2.12.

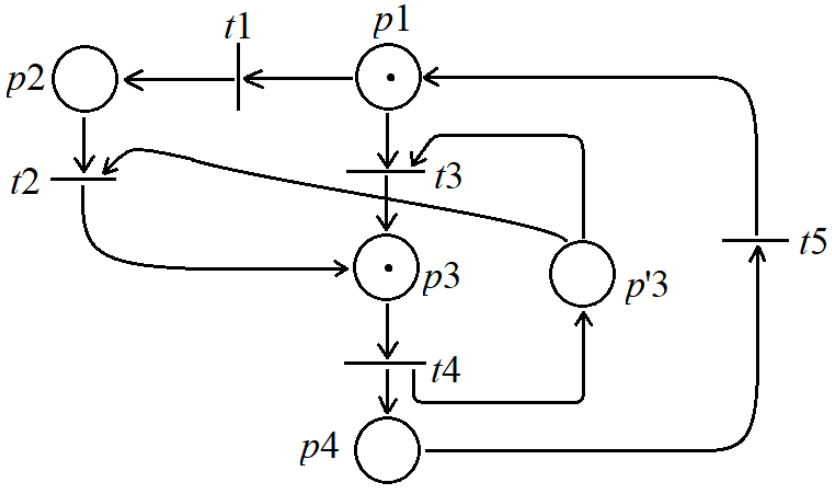


Fig. 2 Rețeaua Petri cu capacitate infinită corespunzătoare celei din fig. 2.11

2.6. Graf de accesibilitate

Urmărind executarea secvențială a tranzițiilor, se pot găsi marcajele succesive ale rețelei, proces descriptibil într-o manieră sintetică printr-un graf (diferit de graful rețelei Petri!), ce poartă denumirea de graf de accesibilitate (Reachability graph). Graful de accesibilitate al rețelei din fig. 2.10 (a) este prezentat în fig. 2.10 (b).

Pentru marcajul $M_0 = (1\ 0)$ considerat, singura tranziție validată este t_1 (ca tranziție sursă). După executarea lui t_1, se obține $M_1 = (2\ 0)$ și tranzițiile t_2 și t_3 devin validate. În această situație se poate executa fie t_2, fie t_3, executarea uneia dintre ele atrăgând invalidarea celeilalte (nu orice tranziție validată se execută în mod obligatoriu!).

Dacă se execută t_2, marcajul devine $M_2 = (0\ 0)$; dacă se execută t_3, marcajul devine $M_3 = (0\ 1)$ etc.

În reţeaua transformată (N', M'_0) se aplică regula simplă a tranziţiei. Se recomandă drept exerciţiu, construirea grafului de acesibilitate al reţelei (N', M'_0). Acesta este prezentat în fig. 2.10 (d), putându-se constanta că el este *izomorf* cu cel al reţelei iniţiale (desenat în fig. 2.10 (b)). Cu alte cuvinte, cele două reţele (N, M_0) şi respetiv (N', M'_0) sunt echivalente comportamental, ambele prezentând aceleaşi secvenţe de executare a tranziţiilor.

În fapt, exemplul a ilustrat următoarea teoremă:

Teoremă

Fie (N, M_0) o reţea pură, cu capacitate finită, căreia i se aplică regula strictă a tranziţiei. Fie (N', M'_0) reţeaua obţinută prin transformarea lui (N, M_0), utilizând metoda poziţiei complementare. În (N', M'_0) se aplică regula simplă a tranziţiei. Reţelele (N, M_0) şi (N', M'_0) sunt echivalente în sensul ca ambele posedă aceeaşi mulţime de secvenţe posibile pentru executarea tranziţiilor.

În baza teoremei, se constată că utilizarea regulii stricte a tranziției într-o rețea cu capacitatea finită, poate fi întotdeauna înlocuită prin utilizarea regulii simple a tranziției într-o rețea cu capacitate infinită.

În consecință materialul de față este organizat în *termenii rețelelor cu capacitate infinită* cu folosirea *regulii simple a tranziției* . Acest mod de prezentare este de fapt cel întâlnit în majoritatea aplicațiilor.

Îndeplinirea condiției din teorema care cere ca rețeaua să fie pură nu este critică, fiindcă orice buclă autonomă poate fi transformată într-o buclă neautonomă.

2.7. Structuri tipice utilizate în modelarea cu rețele Petri

Se vor prezenta structurile cu rol tipic utilizate în modelarea cu rețele Petri.

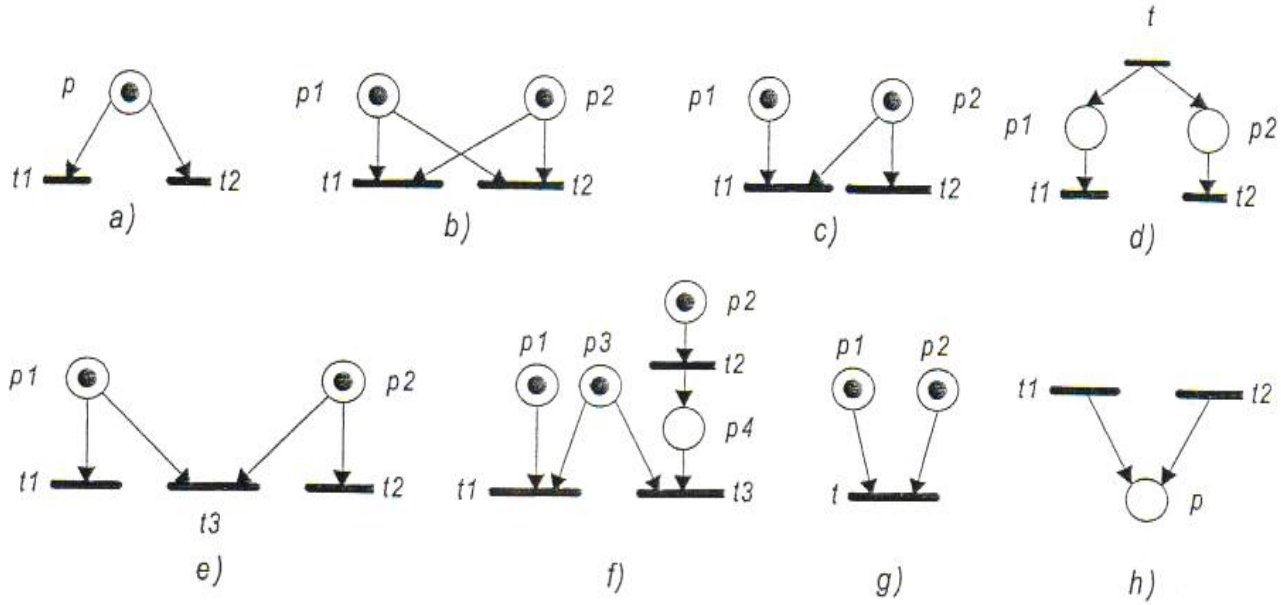


Fig. 2.13 Structuri tipice utilizate în modelarea cu rețele Petri

a) Conflict, decizie sau alegere liberă; b) Alegere liberă extinsă; c) Alegere asimetrică;

d) Paralelism sau concurență; e) Concluzie simetrică; f) Confuzie asimetrică;

g) Sincronizare; h) Post-condiție comună.

Structurile tipice utilizate în modelarea cu rețele Petri sunt:

a) Conflict, decizie sau alegere liberă;
b) Alegere liberă extinsă;
c) Alegere asimetrică;
d) Paralelism sau concurență;
e) Confuzie simetrică;
f) Confuzie asimetrică;
g) Sincronizare
h) Post-condiție comună.

a) Conflict, decizie sau alegere liberă

Două evenimente e_1 și e_2 sunt în conflict (fig. 2.13 a) dacă realizarea lor nu se poate efectua în același timp. În momentul executării oricărei dintre tranzițiile t_1 sau t_2, jetonul va părăsi poziția p, ducând la invalidarea tranziției care nu a fost executată. În fapt, ambele tranziții sunt validate, dar numai una dintre ele se poate executa, execuția uneia conducând la invalidarea celei de-a doua.

O astfel de structură se poate folosi în modelarea unui proces de luare a unei decizii condiționând realizarea unei tranziții sau în cazul unei alegeri libere. Din acest motiv, o astfel de structură poate fi numită conflict, decizie sau alegere liberă, în funcție de modul în care este interpretată.

b) Alegerea liberă extinsă

Atât tranziția t_1 cât și tranziția t_2 sunt validate, dar numai una dintre ele se va executa. În urma executării oricărei dintre cele două tranziții (dar numai a uneia), jetoanele din cele două poziții p_1 și p_2 vor fi consumate ducând la invalidarea tranziției care nu este executată. Alegerea tranziției care va fi executată nu depinde de nici o condiție, aceasta realizându-se în mod aleator. Astfel, se realizează o alegere liberă extinsă.

c) Alegere asimetrică

În acest caz, atât tranziția t_1 cât și tranziția t_2 sunt validate. Executarea oricărei dintre aceste tranziții conduce la invalidarea celeilalte. Tranziția t_1 se execută doar dacă avem jeton atât în poziția p_1 cât și în poziția p_2, pe când tranziția t_2 se execută doar dacă în poziția p_2 avem jeton, nedepinzând de starea poziției p_1. Deoarece executarea tranziției t_1 este condiționată suplimentar de prezența unui jeton în poziția p_1, structura tipică se numește alegere asimetrică.

d) Paralelism sau concurență

Executarea tranziției t conduce la plasarea unui jeton în fiecare dintre pozițiile p_1 și p_2, astfel devenind validate în același timp tranzițiile t_1 și t_2. Activitățile modelate de ramura poziției p_1, respectiv p_2 se pot desfășura în paralel, iar tranzițiile t_1 și t_2 se execută independent, modelând două evenimente concurente, fără a fi generat un conflict.

e) Confuzia simetrică

Structura tipică caracteristică confuziei prezentată în fig. 2.13 (e), este o îmbinare între concurență și conflict. Tranzițiile t_1 și t_2 sunt validate și pot fi executate în același timp, execuția lor nefiind condiționată de nici o poziție comună. Tranzițiile t_1 și t_2 sunt deci concurente. Tranziția t_3 poate fi executată doar în cazul în care atât poziția p_1 cât și poziția p_2 au câte un jeton. Executarea tranziției t_3 duce la invalidarea tranzițiilor t_1 și t_2, generându-se astfel un conflict. Varianta structurii considerate corespunde confuziei simetrice.

f) Confuzia asimetrică

Tranzițiile t_1 și t_2 pot fi executate în același timp. Tranziția t_1 este validată și poate fi executată, nefiind condiționată decât de prezența jetoanelor în pozițiile p_1 și p_3 . Tranziția t_2 se execută independent de celelalte tranziții, fiind condiționată doar de existența unui jeton în poziția p_2. Deci tranzițiile t_1 și t_2 sunt concurente. Tranziția t_3 se execută doar dacă tranziția t_1 nu se execută și dacă tranziția t_2 se execută. Aceasta este în conflict cu tranziția t_1 și de asemenea, este condiționată de realizarea tranziției t_2.

g) Sincronizarea

Tranziția t_1 se realizează doar dacă atât în poziția p_1, cât și în poziția p_2 există câte un jeton. Cele doua poziții corespund la două activități care se pot desfășura în paralel, dar a căror finalizare trebuie corelată.

h) Post-condiție comună

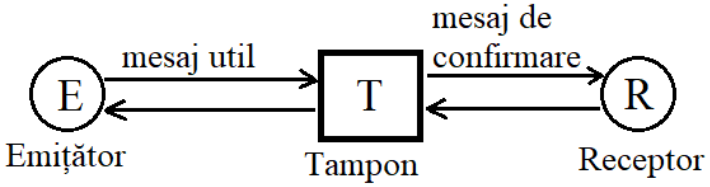
Tranzițiile t_1 și t_2 se pot realiza independent, fiindu-le asociate două evenimente distincte. Realizarea oricăreia dintre ele duce la satisfacerea condiției asociate poziției p. În poziția p va fi plasat câte un jeton după executarea oricăreia dintre tranzițiile t_1 și t_2 .

2.8. Modelarea cu rețele Petri a unui protocol de comunicare

Sincronizarea activităților la emisia, respectiv recepția de mesaje, se bazează, de regulă, pe utilizarea unui tampon (buffer) T, în care emițătorul E depune mesajul util, iar receptorul R, după recepție, depune un mesaj de confirmare (acknowledge).

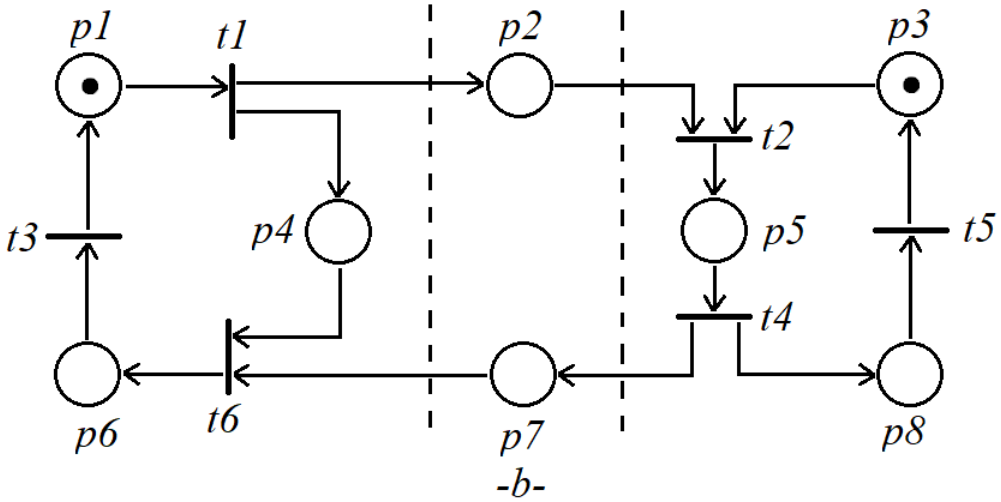
Modul de operare al unui protocol de comunicație simplu, cu transmitere unidirecțională de mesaj util de la E la R, este prezentat în fig. 2.14 a.

Mesajul util pregătit de E este depus în tamponul T și este preluat de R numai atunci când acesta este pregătit să-l recepționeze. După emiterea mesajului util, E așteaptă un mesaj de confirmare, pe care R îl depune în T, de îndată ce a încheiat de recepționat mesajul util. După recepționarea mesajului de confirmare, E pregătește un nou mesaj util pentru transmitere. După recepționarea mesajului util, R procesează mesajul util curent și se pregătește să primească un nou mesaj util. Funcționarea descrisă este ciclică.



-a-

Modelul de tip rețea Petri netemporizată a acestui protocol este reprezentat în fig. b.



-b-

Fig. 2.14 *Modelul de tip Petri al unui protocol de comunicații.*

Semnificațiile pozițiilor din modelul prezentat în fig. 2.14

Poziție	Condiție
p1	Emițătorul (E) pregătește mesajul.
p2	Buffer-ul de comunicație (T) conține mesajul.
p3	Receptorul (R) este pregătit pentru recepție.
p4	(E) așteaptă semnalul de confirmare (*acknowledge* – ACK).
p5	(R) primește mesajul.
p6	(E) primește semnalul de confirmare (ACK).
p7	(T) conține semnalul de confirmare (ACK).
p8	(R) procesează mesajul primit.

Semnificațiile tranzițiilor din modelul prezentat în fig. 2 14

Tranziție	Eveniment
t1	(E) începe transmisia mesajului.
t2	(R) începe primirea mesajului.
t3	(E) termină de primit semnalul de confirmare (ACK) și începe pregătirea unui nou mesaj.
t4	(R) termină primirea mesajului și începe transmisia semnalului de confirmare (ACK) și procesarea mesajului.
t5	(R) termină procesarea și începe pregătirea pentru recepționarea unui nou mesaj.
t6	(E) începe primirea semnalului de confirmare (ACK).

1) Succesiunea de executări de tranziții care, plecând din marcajul initial din fig. 2.14 -b- asigură revenirea la acest marcaj, este reprezentată de arborele de accesibilitate din fig. 2.15, împreună cu specificarea marcajelor.

Se observă că, plecând din marcajul inițial M_0 succesiunea de executări de tranziții care asigură revenirea la acesta este: $\sigma = t_1\ t_2\ t_4\ t_5\ t_6\ t_3$.

$M_0 = [p_1, p_2, p_3, p_4, p_5, p_6, p_7, p_8]$

$M_0 = [1, 0, 1, 0, 0, 0, 0, 0]$

$M_1 = [0, 1, 1, 1, 0, 0, 0, 0]$

$M_2 = [0, 0, 0, 1, 1, 0, 0, 0]$

$M_3 = [0, 0, 0, 1, 0, 0, 1, 1]$

$M_4 = [0, 0, 1, 1, 0, 0, 1, 0]$

$M_5 = [0, 0, 0, 0, 0, 1, 0, 1]$

$M_6 = [0, 0, 1, 0, 0, 1, 0, 0] - t_5\ (M_5)$

$M_7 = [1, 0, 0, 0, 0, 0, 0, 1] - t_3\ (M_5)$

$M_0 = [0, 1, 0, 1, 0, 0, 0, 1]$

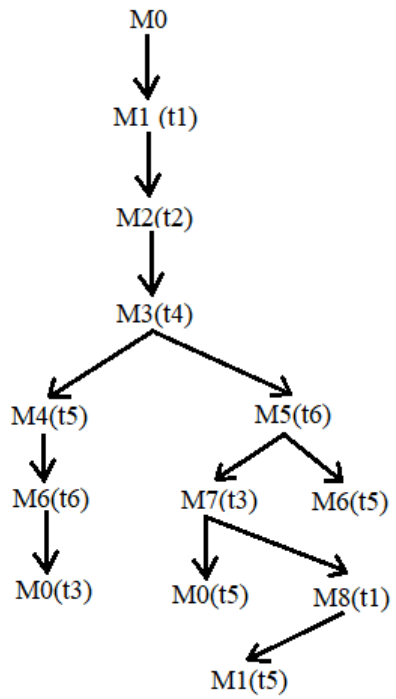


Fig. 2.15 Arborele de accesibilitate pentru rețeaua Petri din fig. 2.14 -b-.

2) Pe baza semnificației fizice a tranzițiilor din modelul studiat, succesiunii de tranziții σ, determinate la punctul precedat, îi corespunde următoarea succesiune de evenimente care au loc la transmiterea completă a unui mesaj:
   - (E) începe transmisia mesajului (t_1);
   - începe primirea semnalului de configurare (ACK) (t_2);
   - (R) începe primirea mesajului (t_4);
   - (R) termină primirea mesajului și începe transmisia semnalului ACK și procesarea semnalului (t_5);
   - (R) termină procesarea și începe pregătirea pentru recepționarea unui nou mesaj (t_1);
   - (E) termină de primit semnalul ACK și începe pregătirea unui nou mesaj (t_3).

Observație: Se poate observa că, în afară de secvența σ precizată anterior, mai există secvențe de executări de tranziții care, plecând din marcajul inițial, asigură revenirea la acesta, și anume:

$$\sigma = t_1\ t_2\ t_4\ t_6\ t_3\ t_5.$$

Deoarece duratele operațiilor care au loc în sistemul fizic nu sunt implicate în modelul logic reprezentat de rețeaua Petri netemporizată, nu se poate preciza ordinea în care se vor executa tranzițiile $t_5\ t_6$.

2.9. Subclase de rețele Petri ordinare

Rețelele Petri ordinare pot fi organizate în *subclase* definite pe baza unor considerente *topologice*. Apartenența unei rețele la o anumită subclasă este reflectată în *capacitatea de modelare*, în sensul că rețeaua nu va putea conține toate *structurile tipice* definite în secțiunea precedentă, ci numai o parte din acestea (în funcție de subclasa căreia îi aparține). Această partajare pe subclase servește la formularea unor *tehnici dedicate* analizei anumitor proprietăți comportamentale pentru aceste tipuri de rețele.

Mașina de stare (*state machine*) este o rețea Petri ordinară în care fiecare tranziție t are o singură poziție de intrare și o singură poziție de ieșire. Formalizând:

$$\forall t \in T: |{}^*t| = |t^*| = 1,$$

unde |*| notează cardinalitatea (numărul de elemente al) mulțimii *.

Referindu-ne la structurile tipice, se constată că o mașină de stare *conține numai* alegeri libere și post-condiții commune.

Graful marcat (*marked graph*) sau ***graful de evenimente*** (*event graph*) este o rețea Petri ordinară în care fiecare poziție p are o singură tranziție de intrare și o singură tranziție de ieșire. Formalizând:

$$\forall p \in P: |{}^*p| = |p^*| = 1.$$

În termenii structurilor tipice, se constată că un graf marcat *conține numai* concurențe și sincronizări.

Termenul de graf marcat sau graf de evenimente se datorează faptului că acest tip de rețea Petri poate fi reprezentată printr-un *graf orientat unipartit* (care posedă un singur tip de noduri!) în care *nodurile* corespund tranzițiilor (adică evenimentelor), *arcele* corespund pozițiilor, iar jetoanele se plasează *pe arce*(adică arcele sunt marcate).

Astfel regula tranziției se aplică nodurilor grafului orientat, o executare constând, în această reprezentare grafică, în îndepărtarea unui jeton de pe fiecare din arcele de intrare ale nodului în cauză și adăugarea unui jeton pe fiecare din arcele de ieșire.

Exemplu:

Rețeaua Petri prezentată în fig. 2.14 -b- face parte din clasa grafurilor marcate. Datorită tipului particular de rețea, drept reprezentare grafică se poate utiliza un graf unipartit, conform fig. 2.16.

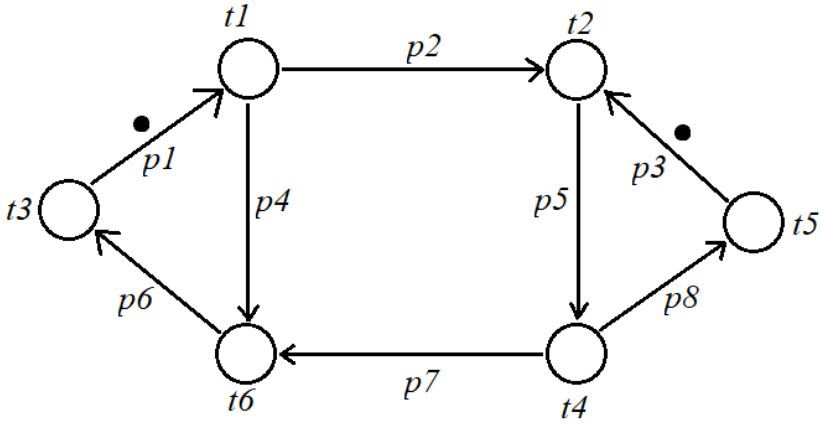


Fig. 2.16 *Modelul de tip graf marcat al protocolului de comunicații reprezentat sub formă de graf orientat unipartit.*

Structurile formate din t_1, p_4, p_2, *și respectiv* t_4, p_8, p_7 reprezintă structuri de tip paralelism sau concurență.

Cu prima se modelează faptul că transmiterea mesajului util (producerea evenimentului t_1), conduce simultan la schimbarea stării lui E și T (p_4 *și* p_2 primesc câte un jeton).

Similar, se justifică și utilizarea celei de a doua structuri, de tip paralelism, care se referă la schimbarea stărilor lui R și T.

Pe de altă parte, structurile formate din p_4, p_7, t_6 și respectiv p_3, p_2, t_2, reprezintă structuri de tip sincronizare.

Cu cea de a doua, se modelează faptul că începerea recepționării mesajului util (producerea evenimentului t_2) este posibilă numai dacă T conține un mesaj și R este gata de recepție (p_2 și p_3 conțin câte un jeton). Similar se justifică și utilizarea primei structuri de tip sincronizare.

3. AUTOMATE

Se urmărește abordarea problematicii construirii modelelor netemporizate pentru procesele pilotate de evenimente, prin prisma posibilității de a utiliza doua tipuri de reprezentări**:** rețelele Petri si automatele.

Un *automat (automaton)* este un dispozitiv care este capabil să implementeze un limbaj în acord cu reguli bine definite.

Cel mai simplu mod de a prezenta noțiunea de automat este de a considera reprezentarea grafică a lui, sau diagrama de tranziție a stărilor.

Se consideră un exemplu de automat simplu reprezentat de graful orientat sau diagrama de tranziție a stărilor din fig. 3.1:

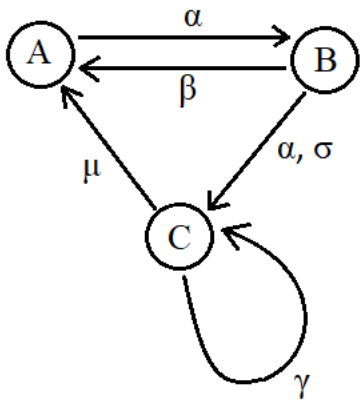


Fig. 3.1.

Nodurile *reprezintă* ***stările*** *și* ***arcele*** *etichetate* ***tranzițiile*** *dintre diverse noduri* (*stări*).
Acest graf permite o descriere completă a automatului.
Mulțimea nodurilor este chiar mulțimea stărilor automatului

$$X=\{A, B, C\}.$$

Mulțimea (setul) etichetelor tranzițiilor arcelor formează mulțimea evenimentelor (alfabetul) automatului

$$E=\{\alpha, \beta, \gamma, \sigma, \mu\}.$$

Arcele din graf realizează reprezentarea grafică a *funcției de tranziție a automatului*, exprimată ca:

- $f: X \times E \rightarrow X$;
- $f(A, \alpha)=B$; $f(B, \alpha)=f(B, \sigma)=C$; $f(C, \mu)=A$;
- $f(B, \beta)=A$; $f(C, \gamma)=C$.

Notația f(A, α)=B înseamnă că dacă automatul este în starea A, atunci la apariția evenimentului α, automatul va realiza o tranziție instantanee în starea B.

Cauza apariției evenimentului α este irelevantă, evenimentul poate fi o intrare externă a sistemului modelat de automat sau poate fi un eveniment spontan („generat" de sistemul modelat prin automat).

Se observă că un eveniment poate apărea fără a schimba starea ca de exemplu f(C, γ)=C.

De asemenea, doua evenimente distincte pot apărea la o stare dată cauzând exact aceeași tranziție, ca de exemplu f(B, α)=f(B, σ)=C. Nu se poate face o distincție între evenimentele α și γ din simpla observare a tranziției din starea B în starea C.

Se constată că din fiecare nod (stare) : X={A, B, C}, al grafului orientat pornește un număr de arce orientate egal cu numărul de elemente al lui Γ(x), unde **Γ(x)** reprezintă ***mulțimea evenimentelor posibile***:

$\Gamma(A)=\{\alpha\}$; $\Gamma(B)=\{\alpha, \beta, \gamma\}$; $\Gamma(C)=\{\gamma, \mu\}$.

Definiție:

*Un **automat** (automaton) este un cvintuplu **(E, X, Γ, f, x_0)** în care semnificația parametrilor este următoarea:*

- ***E** este mulțimea evenimentelor (mulțime numărabilă);*
- ***X** este spațiul stărilor(mulțime numărabilă);*
- ***Γ(x)** este mulțimea evenimentelor posibile (care se pot produce) în starea $x \in X$ (privită ca oarecare); $\Gamma(x) \subseteq E$. (Cu alte cuvinte, apariția evenimentelor din E este dependentă de starea curentă a sistemului x);*
- ***f** este funcția de tranziție a stării:*

$f: X \times E \rightarrow X$, $x'=f(x, e)$,

*definită numai pentru $e \in \Gamma(x)$, atunci când starea este x; f(x, e) nu este definită pentru $e \notin \Gamma(x)$. Starea **x'** notează starea în care tranziționează automatul aflat in starea x, ca urmare a producerii evenimentului $e \in \Gamma(x)$. Cu alte cuvinte , problema tranziției dintr-o starea x într-o altă stare x' se pune numai pentru evenimentele posibile în starea x.*

- *$x_0 \in X$ este starea inițială.*

Oricărui automat i se poate atașa și o ieșire, situație când automatul este definit printr-un septuplu (**E, X, Γ, f, x_0, Y, g**), unde E, X, Γ, f, x_0 păstrează semnificațiile anterior comentate, la care se adaugă obiectele definite mai jos:

- **Y** este mulțimea ieșirilor.
- **g** este funcția ieșirii: $g: X \times E \rightarrow Y$, $y=g(x, e)$,

  definită numai pentru $e \in \Gamma(x)$, atunci când starea este x;
  g(x, e) nu este definită pentru $e \notin \Gamma(x)$.

Formalizarea matematică prezentată anterior îi corespunde o reprezentare grafică de tip graf orientat, denumită diagrama tranzițiilor. În cazul când este definită și ieșirea automatului, arcul orientat pornește din nodul x și corespunde evenimentului $e \in \Gamma(x)$ va primi o a doua etichetă cu valoarea ieșirii y=g(x, e), separată prin virgulă de etichetă ‚e'.

3.1. Transformări între modelele tip automat și tip rețele Petri

În exemplul din fig. 2.10 s-a atașat unei rețele Petri un graf de accesibilitate care transpune mecanismul de funcționare al rețelei Petri într-o descriere de tip automat: nodurile grafului orientat reprezintă stările sistemului (adică marcajul rețelei), iar fiecare arc orientat este etichetat cu evenimentul care produce schimbarea de stare corespunzătoare (adică executarea unei tranziții a rețelei). Analizând, intuitiv, modul cum s-a realizat asocierea automatului, fără a intra în detalii tehnice de demonstrație, se constată că *pentru orice rețea Petri se poate construi un automat care pune în evidență aceeași dinamică pilotată de evenimente*. O asemenea construcție este posibilă și în cazul când rețeaua Petri posedă un număr infinit de stări (datorită creșterii nelimitate a marcajului anumitor poziții).

Totodată este adevărată și afirmația inversă, adică , *dat fiind un automat, se poate construi o rețea Petri care pune în evidență aceeași dinamică pilotată de evenimente*.

Un procedeu care permite construirea unei rețele Petri (P, T, F, W, M_0), plecând de la automatul (E, X, Γ, f, x_0) se bazează pe următorul **algoritm**.

Fiecare stare din X devine o poziție în P, adică P=X. Apoi fiecărei perechi de stări din X, desemnată prin (x, x'), cu x'=f(x, e), e∈Γ(x), i se asociază o tranziție t∈T în rețeaua Petri, atașând și arcele (x, t), (t, x')∈F, ambele cu ponderea 1.

Prin această construcție, unui același eveniment $e \in E$, i se pot asocia mai multe tranziții în T.

Exemplu:

Se consideră automatul din fig. 3.2 a. Se construiește rețeaua Petri conform algoritmului discutat. Astfel $P=\{p_1, p_2, p_3\}$, $p_1=\{A\}$, $p_2=\{B\}$, $p_3=\{C\}$. Pentru determinarea mulțimii T se examinează, pe rând, tranzițiile posibile din cele 3 stări, construind perechile de tipul (x, x'):

- pentru p_1: (p_1, p_2), $p_2=f(p_1, t_1)$, $t_1=\{\alpha\}$;
- pentru p_2: (p_2, p_1), $p_1=f(p_2, t_2)$, $t_2=\{\beta\}$;
  (p_2, p_3), $p_3=f(p_2, t_3)$, $t_3=\{\lambda\}$;
- pentru p_3: (p_3, p_1), $p_1=f(p_3, t_4)$, $t_4=\{\mu\}$.

Se atașează și arcele aferente de pondere 1 și se obține rețeaua Petri din fig. b.

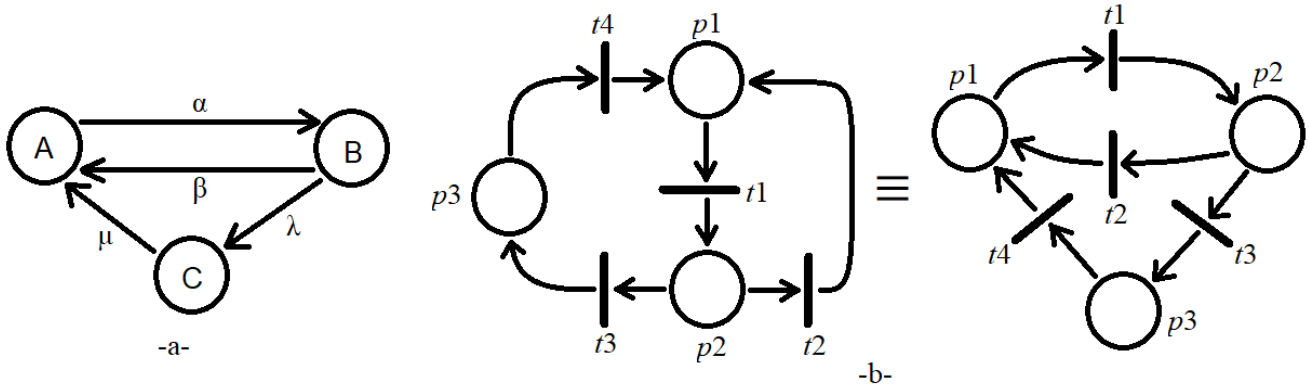


Fig. 3.2.

Se observă că ansamblul format din poziția p_2 și tranzițiile t_2, t_3 reprezintă o alegere liberă, corespunzând apariției unuia din evenimentele β (executarea lui t_2) sau λ (executarea lui t_3).

Procedeul schițat și ilustrat mai sus nu este unicul mod de a asocia, unui automat, o rețea Petri. Mai mult, acest procedeu poate fi caracterizat drept unul formal, în sensul că permite o aplicare directă, fără a necesita investigarea comportării sistemului fizic, care, prin modelare, a condus la descrierea de tip automat. În practică, o atare investigare este întotdeauna recomandată, întrucât pune în evidență aspecte ce pot simplifica structura rețelei Petri.

Sub aspect teoretic, modelul de tip rețea Petri este echivalent cu modelul de tip automat. Aceasta înseamnă că pornind de la un sistem dinamic cu evenimente discrete real, cu semnificație fizică, se pot construi ambele tipuri de modele. Din punct de vedere practic, există însă unele aspecte ce pot justifica faptul că unul din aceste două tipuri de modele este mai profitabil pentru studierea comportării sistemului.

În general, opțiunea unui anumit tip de model rămâne la latitudinea modelatorului, depinzând de experiența și abilitățile acestuia. Există totuși unele *criterii generale, orientative,* care trebuie înțelese în contextul unor situații frecvent întâlnite.

Recomandări pentru alegerea tipului de model

- Utilizarea modelelor tip automat este recomandată atunci când se iau în considerare numai evenimente ce se petrec în universul exterior sistemului fizic care se modelează. Dacă interesează a pune în evidență, prin model, și evenimente interne sistemului fizic, este preferabil să se facă apel la rețelele Petri.
- În cazul când modelul trebuie să permită operarea cu un număr mare de stări (evident distincte), rețelele Petri pot fi mai avantajoase, întrucât diferențierea între stări se realizează prin marcaj și nu prin expandarea topologiei (situație inevitabilă în cazul diagramelor de tranziții asociate automatelor cu un număr mare de stări). Cu alte cuvinte, efortul de modelare în cazul discutat este (în general) mai mic, dacă se caută o topologie de rețea Petri relativ simplă și formularea mecanismului aferent pentru asignarea și modificarea marcajului, decât dacă se urmărește generarea unei diagrame de tranziții cu topologie relativ complexă și complicată.
- Un alt aspect se referă la posibilitatea asamblării unui model din submodele. În principiu, utilizarea rețelelor Petri permite aplicarea cu succes a acestei metodologii de modelare, conferind o flexibilitate sporită în surprinderea detaliilor și rafinarea modelului.

Utilizarea automatelor este mai rigidă, deoarece modelul trebuie privit de la început în ansamblul său, modularizarea fiind greoaie.

4. PROPRIETĂȚI COMPORTAMENTALE ALE REȚELELOR PETRI

Proprietățile comportamentale ale rețelelor Petri netemporizate sunt dependente atât de topologia cât și de marcajul inițial al rețelei, spre deosebire de proprietățile structurale, care iau în considerare numai topologia rețelei, fiind independente de marcajul inițial al acesteia.

Accesibilitate (reachable)

O secvență de executări de tranziții ale unei rețele conduce la modificarea marcajului (distribuției de jetoane), în conformitate cu aplicarea regulii tranziției.

Un marcaj $\boldsymbol{M_n}$ *se spune ca e* ***accesibil*** *din marcajul inițial* M_0*, dacă există o secvență de executări de tranziții, care transformă* M_0 *în* M_n*.*

Această secvență de executări (de tranziții) se notează prin: $\boldsymbol{\sigma = M_0\, t_{i1}\, M_1\, t_{i2}\, M_2 \ldots t_{ik-1}\, M_k}$, sau, simplu, prin: $\boldsymbol{\sigma = t_{i1}\, t_{i2} \ldots t_{ik}}$, când nu interesează succesiunea de marcaje.

Faptul că marcajul M_n este accesibil din M_0 prin secvența de executări σ se notează $\boldsymbol{M_0[\sigma > M_n]}$.

Mulțimea tuturor marcajelor care pot fi atinse în rețeaua (N, M_0), pornind din M_0, se notează prin $\boldsymbol{R(N, M_0)}$, sau, simplu, prin $\boldsymbol{R(M_0)}$, atunci când se subînțelege rețeaua N la care ne referim.

Mulțimea tuturor secvențelor de executare posibile în rețeaua (N, M_0), pornind din M_0, se notează prin L(N, M_0) sau, simplu, prin L(M_0) atunci când se subînțelege rețeaua N la care ne referim.

Studierea accesibilității constă în a decide dacă un anumit marcaj M_n aparține sau nu mulțimii R(N, M_0).

Mărginire (bounded)

O rețea Petri (N, M_0*) se spune că este* ***k-mărginită*** *sau, pe scurt,* ***mărginită*** *dacă numărul de jetoane din fiecare poziție nu depășește un număr finit* **k** *pentru orice marcaj ce este accesibil din starea* M_0 *(adică pentru orice secvență de executări de tranziții, pornind de la marcajul* M_0*).*

În limbaj matematic, aceasta revine la $\boldsymbol{M(p) \leq k}$ pentru orice p și orice $M \in R(M_0)$.

O rețea Petri (N, M_0) se spune că este ***sigură*** dacă ea ***este 1-mărginită***.

Din punct de vedere practic, când rețeaua modelează un proces, proprietatea de mărginire permite a studia eventualele depășiri ale unor capacități fizice de procesare/memorare a informației sau de prelucrare/stocare a produselor. Mărginirea asigură nedepășirea anumitor valori, indiferent de secvența de evenimente (adică tranziții executate).

Exemplul 4.1

Se consideră rețeaua Petri din fig. 4.1 (a), pentru care fig. (b) prezintă graful de accesibilitate (exemplul dat în cadrul metodei pozițiilor complementare). Toate marcajele accesibile pornind de la marcajul inițial apar în acest graf.

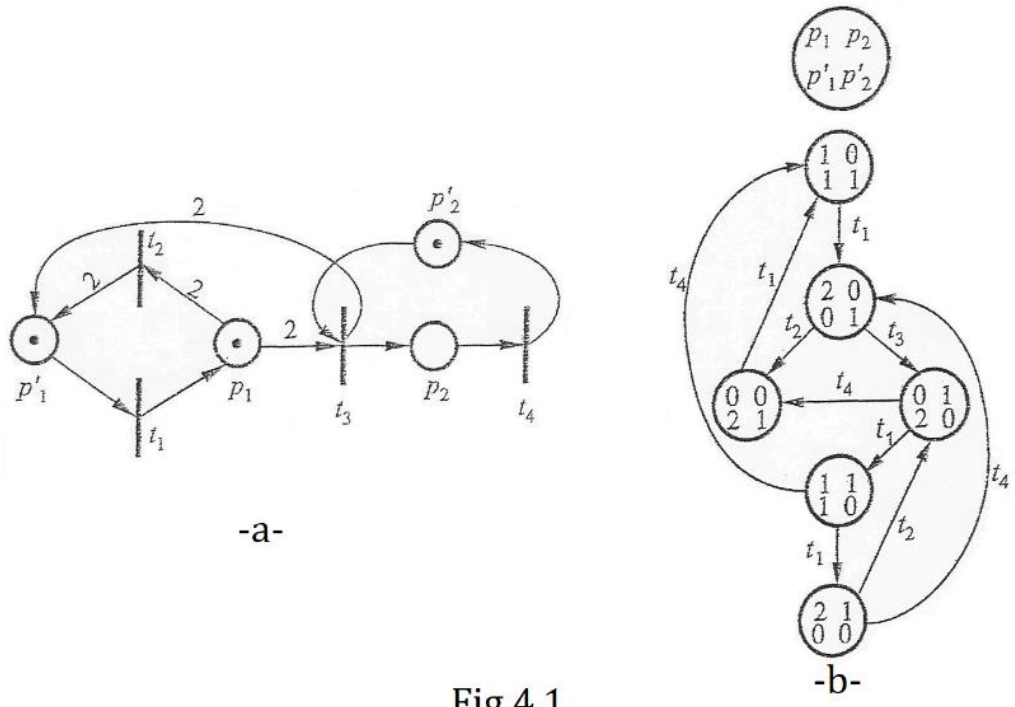


Fig 4.1

a) Rețeaua (N', M'$_0$) cu capacitate infinită;
b) Graful de accesibilitate al rețelei (N', M'$_0$) obținut prin aplicarea regulii simple a tranziției .

Se constată că există marcaje care nu pot fi atinse pornind de la marcajul inițial, și anume cele pentru care există $M(p_1)+M'(p_{1'})\neq 2$ și/sau $M(p_2)+M(p_{2'})\neq 1$.

Reţeaua considerată este 2-marginită, numărul maxim de jetoane ce poate fi acumulat de o poziție fiind 2 (a se inspecta graful de accesibilitate).

Viabilitate (live)

O rețea Petri (N, M_0) se spune că este ***viabilă*** *(sau, echivalent, M_0 se spune că este un* ***marcaj viabil*** *pentru N) dacă, indiferent de marcajul care a fost atins pornind de la M_0, este posibil ca, în continuare, să fie executată orice tranzacție t a rețelei*.

Până la executarea lui *t* poate fi necesară, eventual, executarea unui număr *finit* de alte tranziții.

Un marcaj pentru care nici o tranziție a rețelei nu mai poate fi efectuată se numește ***deadlock*** sau ***blocaj***.

În baza definiției viabilității se constată că o *rețea viabilă operează fără deadlock*. Pe de altă parte, o *rețea care nu este viabilă nu evoluează în mod obligatoriu către deadlock*, existând una sau mai multe *tranziții care pot fi executate de o infinitate de ori*.

Din punct de vedere practice, când rețeaua modelează un proces, proprietatea de viabilitate permite a studia funcționarea fără incidente nereparabile (de factură logică, adică nu defecte), care să necesite o intervenție externă procesului.

Întrucât proprietatea de *viabilitate*, în formularea de mai sus, este suficient de restrictivă, s-a căutat relaxarea ei, introducându-se mai multe ***grade de viabilitate***, după cum urmează.

O tranziție *t* din rețeaua Petri (N, M_0) se spune că este:

0^0 viabilă L_0 *(sau **blocată**), dacă t nu se mai poate executa în nici o secvență din $L(M_0)$.*

1^0 viabilă L_1 *(potențial executabilă), dacă t se mai poate executa cel puțin o dată în unele secvențe din $L(M_0)$.*

2^0 viabilă L_2*, dacă, pentru orice întreg k>0, t se mai poate executa de cel puțin k ori în unele secvențe din $L(M_0)$.*

3^0 viabilă L_3*, dacă t se mai poate executa de un număr finit de ori pentru unele secvențe din $L(M_0)$.*

4^0 viabilă L_4 *(sau viabilă) dacă t este viabilă L_1 pentru orice secvență din $L(M_0)$.*

*Se spune că o tranziție este viabilă L_k **strict**, dacă ea este viabilă L_k dar nu este viabilă $L_{(k+1)}$, pentru k=1, 2, 3.*

O rețea Petri (N, M_0) se spune că este ***viabilă L_k***, dacă fiecare tranzacție a ei este viabilă L_k, k=1, 2, 3, 4.

Viabilitatea de tip L_4 a unei rețele este cea mai puternică și corespunde definiției proprietății de viabilitate ce a fost dată la începutul acestui paragraf. La polul opus se află viabilitatea L_0 a unei rețele care coincide cu definiția dată pentru deadlock.

Se constată că între gradele de viabilitate introduse mai sus, există următoarea condiționare logică de tip implicație:

viabilă L_4=> viabilă L_3=> viabilă L_2=> viabilă L_1.

Noțiunea de strictă viabilitate L_k se aplică la o rețea Petri (totalitatea tranzițiilor), păstrând semnificația introdusă anterior pentru o singură tranziție.

Exemplul 4.2

Se consideră rețeaua Petri din fig. 4.2 (a). Rețeaua nu este viabilă, întrucât dacă se execută t_1 se ajunge la deadlock, (fig. 4.2 (b)), nici o altă tranziție nemaiputându-se executa.

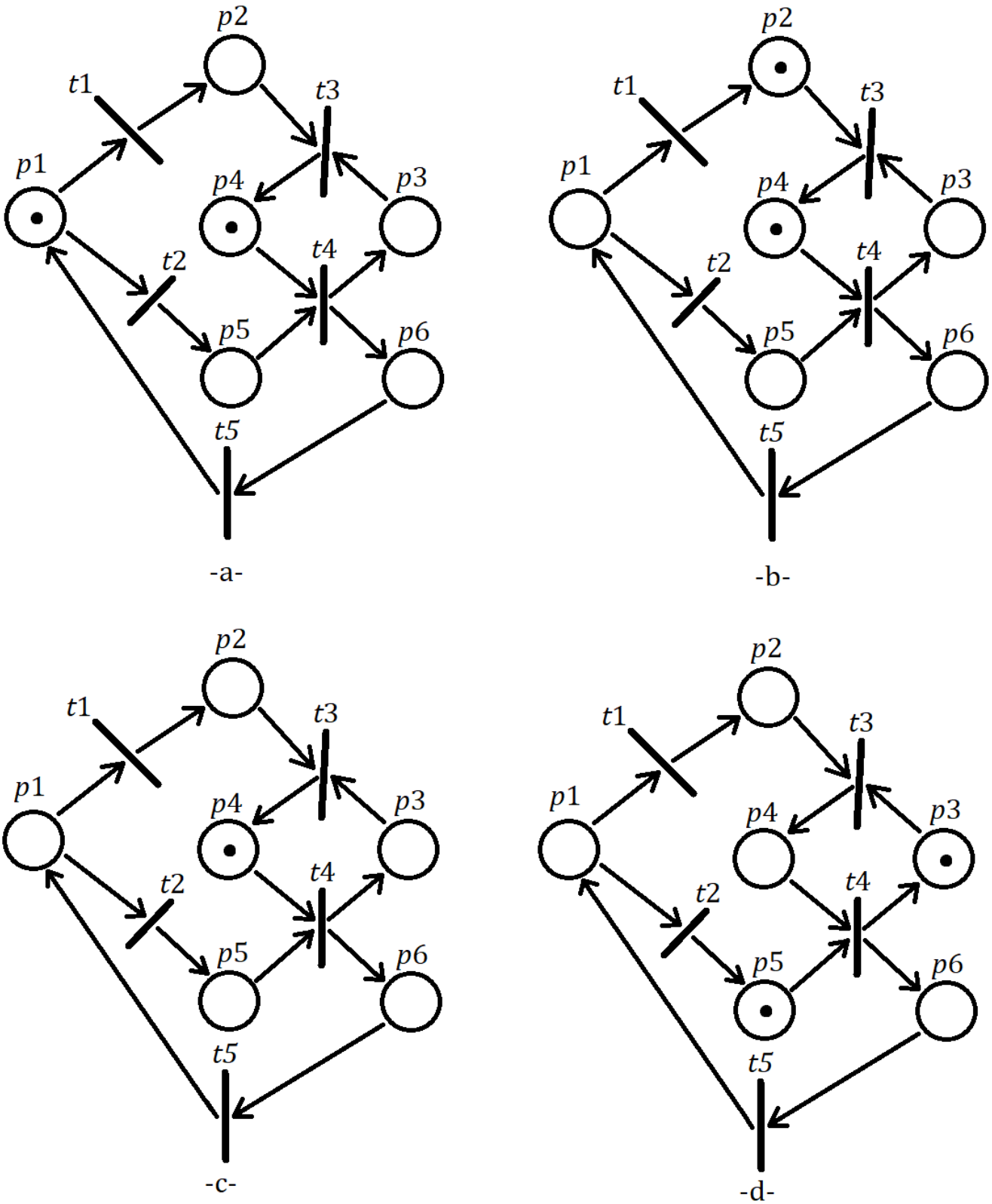


Fig. 4.2

a) Marcajul inițial;

b) Marcajul de deadlock atins după executarea lui t_1;

c) Marcajul de deadlock atins după executarea secvenței t_2, t_4, t_5, t_1, t_3;

d) Marcajul de deadlock atins după executarea secvenței t_2, t_4, t_5, t_2.

Aplicând gradele de viabilitate, se poate afirma că rețeaua este viabilă L_1 strict, deoarece fiecare tranziție poate să mai fie executată exact o singură dată, conform secvenței t_2, t_4, t_5, t_1, t_3. În urma executării acestei secvențe se ajunge la deadlock-ul din fig. 4.2 (c).

Se constată că, deși rețeaua (în totalitate) este viabilă L_1 strict, tranziția t_2 este viabilă L_2 strict. Într-adevăr secvența t_2, t_4, t_5, t_2 permite executarea lui t_2 de două ori, după care se ajunge la deadlock-ul din fig. 4.2 (d).

În fapt, secvențele de executări comentate anterior sunt singurele care pot apărea pentru marcajul inițial din fig. 4.2 (a), toate conducând la deadlock. Totodată se observă că deadlock-urile din fig. 4.2 (b), (c) și (d) corespund la trei marcaje finale diferite.

Exemplul 4.3

Se consideră rețeaua Petri din fig. 4.3 (a). Tranziția t_0 este viabilă L_0 (blocată), neputând fi executată pentru nici o secvență. Se poate afirma deci că rețeaua nu este viabilă.

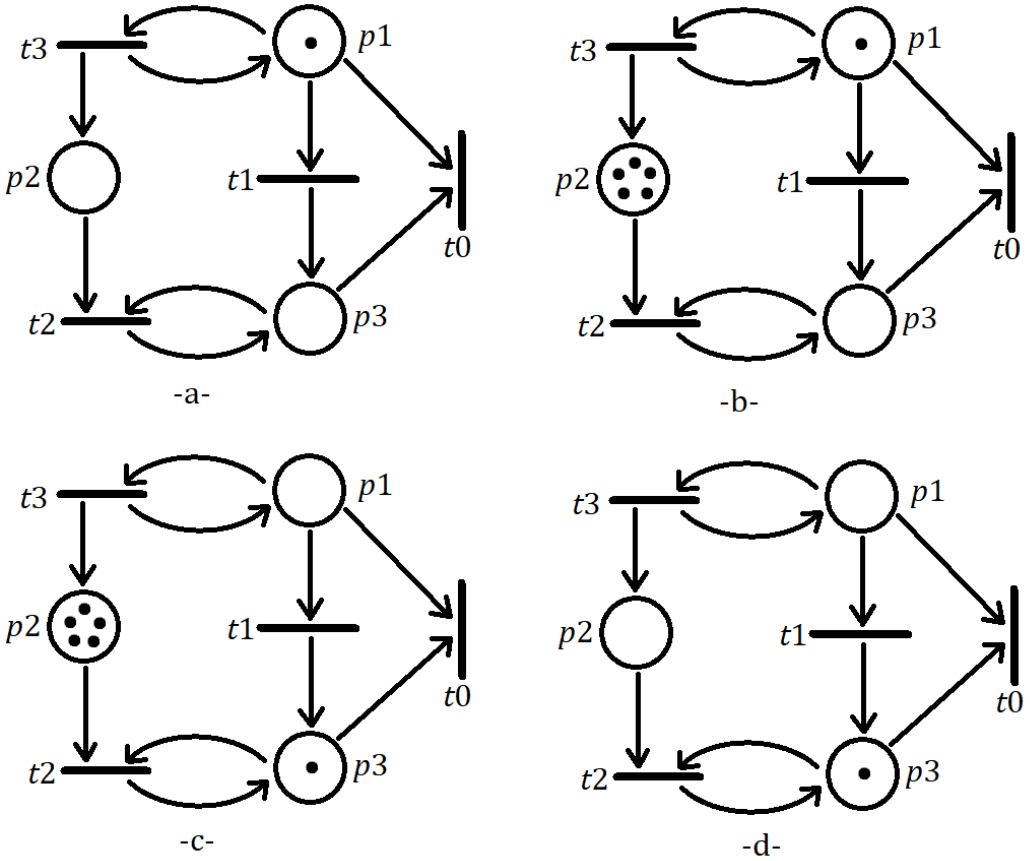


Fig. 4.3

a) Marcajul inițial;

b) Marcajul atins după secvența repetitivă t_3, ..., t_3;

c) Marcajul atins după o secvență (t_3, ..., t_3)t_1;

d) Marcaj de deadlock.

Tranziția t_3 poate fi executată de o infinitate de ori în secvența t_3, t_3, ..., t_3 ... conducând la acumularea de jetoane în p_2, conform fig. 4.3 (b). Deci t_3 este viabilă L_3. (Deocamdată nu se știe dacă t_3 este viabilă L_3 strict sau nu). Se constată că rețeaua nu este viabilă, dar secvența infinită t_3, t_3, ..., t_3, ... nu conduce la blocaj.

Tranziția t_1 nu poate fi executată decât în secvența (t_3, ..., t_3)t_1, conducând la marcajul din fig. 4.3 (c). Așadar t_1 este viabilă L_1 strict.

Dacă se consideră marcajul din fig. 4.3 (c), t_2 poate fi executată de un număr finit de ori (egal cu numărul jetoanelor din p_2), ajungând, în final, la deadlock-ul din fig. 4.3 (d). Deci t_2 este viabilă L_2 strict.

Cum t_3 nu poate fi executată pornind dintr-un marcaj oarecare (vezi marcajele din fig. 4.3 (c) și (d)), rezultă că t_3 nu este viabilă L_4, deci este viabilă L_3 strict.

Reversibilitate

*O rețea Petri (N, M_0) se spune că este **reversibilă**, dacă pentru orice marcaj $M \in R(M_0)$, marcajul inițial M_0 este, la rândul său, accesibil când se pornește din M. Astfel, într-o rețea reversibilă, întotdeauna este posibilă întoarcerea la marcajul inițial.*

Proprietatea de reversabilitate poate fi *relaxată* în sensul că nu se urmărește întoarcerea chiar în marcajul inițial M_0, ci într-un alt marcaj, notat **M'**, care poartă denumirea de ***marcaj (sau stare) recuperabil***. Astfel, se spune că un marcaj M' este recuperabil dacă, pentru orice marcaj $M \in R(M_0)$, M' este accesibil din M.

Din punct de vedere practic, când rețeaua modelează un proces, proprietatea de reversabilitate permite a studia repetabilitatea desfășurării anumitor activități sau a apariției anumitor condiții.

Exemplu:

Se consideră rețeaua Petri din fig. 4.1 (a), având graful de accesibilitate din fig. 4.1 (b). Se constată că rețeaua este reversibilă, marcajul inițial din fig. 4.1 (b) putând fi atins din orice alt marcaj al rețelei. Mai mult, orice marcaj din graful de accesibilitate (din fig. 4.1 (b)) este recuperabil. Alegând un marcaj din acest graf, indiferent din care alt marcaj al grafului pornește, există o secvență finită de executări de tranziții care conduce la marcajul desemnat, ceea ce demonstrează posibilitatea de recuperare a acestuia.

Exemplul 4.4

Fiecare din rețelele Petri din fig. 4.4 posedă numai două din cele trei proprietăți, după cum urmează. Rețeaua din fig. 4.4 (a) nu este mărginită (p_1 este nemărginită), este viabilă și este reversibilă. Rețeaua din fig. 4.4 (b) este mărginită, nu este viabilă (t_1 are viabilitate L_0) și este reversibilă. Rețeaua din fig. 4.4 (c) este mărginită, viabilă și nu este reversibilă.

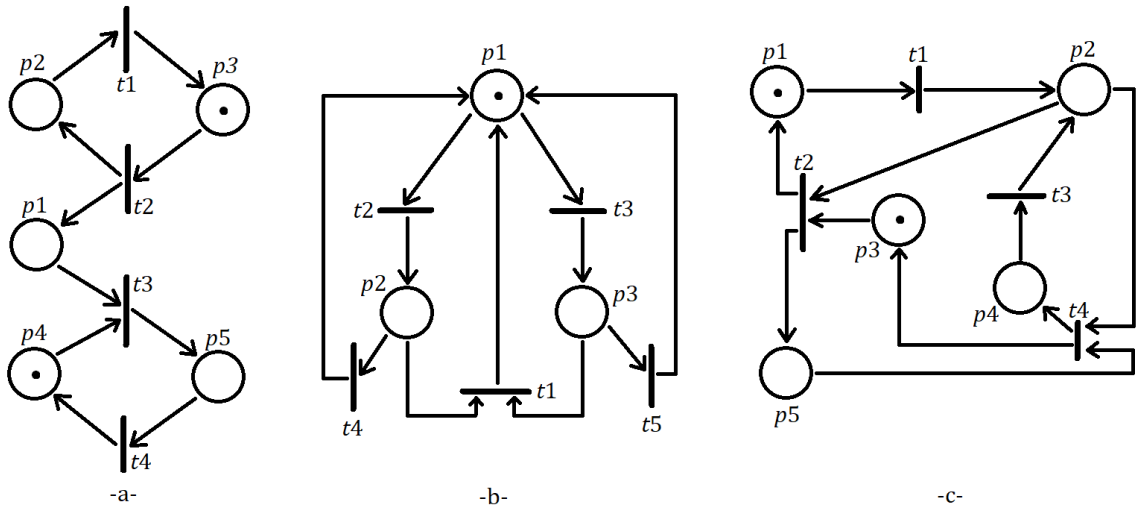


Fig. 4.4 Rețele Petri utilizate în exemplul 4.4:

a) Nemărginită, viabilă, reversibilă;
b) Mărginită, neviabilă, reversibilă;
c) Mărginită, viabilă, nereversibilă.

Exemplul 4.5

Fiecare din rețelele Petri din fig. 4.5 posedă numai câte o proprietate din cele trei proprietăți avute în vedere. Rețeaua din fig. 4.5 (a) nu este mărginită (p_1 este nemărginită), nu este viabilă (t_1 are viabilitate L_0) și este reversibilă.

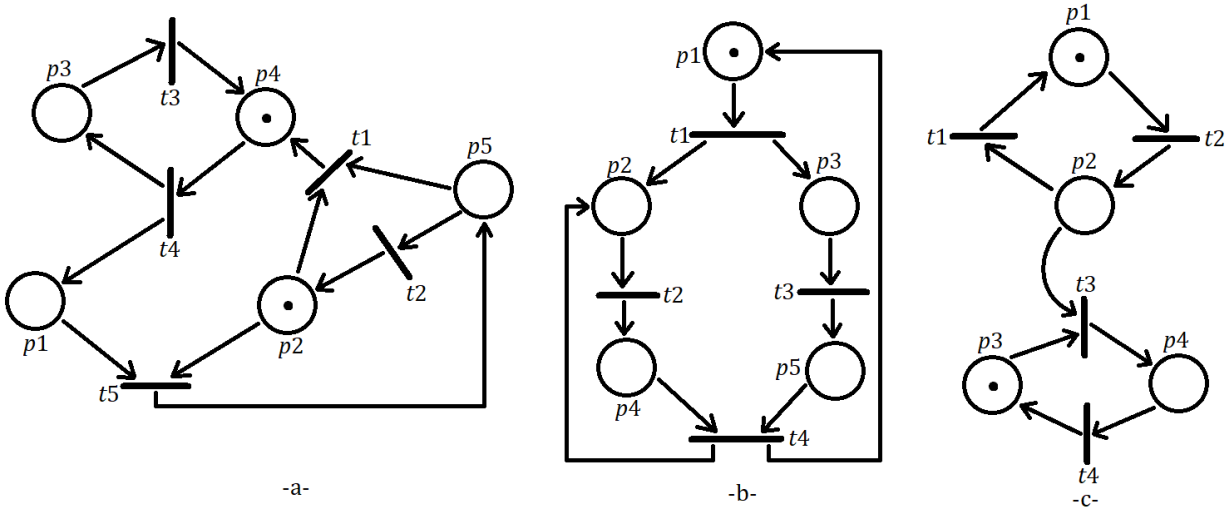


Fig. 4.5 Rețele Petri utilizate în exemplul 4.5

a) Nemărginită, neviabilă, reversibilă;
b) Nemărginită, viabilă, nereversibilă;
c) Mărginită, neviabilă, nereversibilă.

Rețeaua din fig. 4.5 (b) nu este mărginită (p_2 este nemărginită), este viabilă și nu este reversibilă. Rețeaua din fig.4.5 (c) este mărginită, nu este viabilă (t_1, t_2, t_3, t_4 sunt viabile L_3, dar nu sunt viabile L_4) și este nereversibilă.

Exemplul 4.6

În fig. 4.6 se prezintă o rețea Petri care nu posedă nici una din cele trei proprietăți avute în vedere. Rețeaua nu este mărginită (p_1 este nemărginită), nu este viabilă (t_1 este viabilă L_0) și nu este reversibilă.

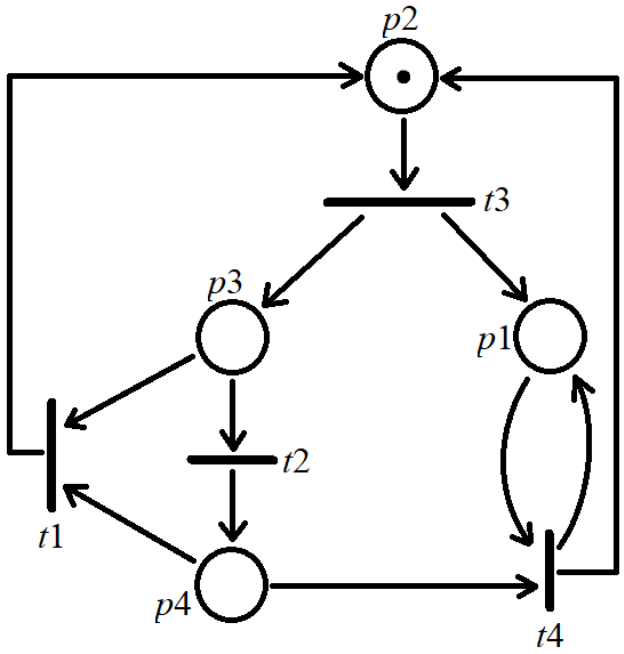


Fig. 4.6 Rețea Petri nemărginită, neviabilă, nereversibilă.

Acoperire

Un marcaj M în rețeaua Petri (N, M_0) se spune că este ***acoperibil (coverable)****, dacă există un marcaj M'∈R(M_0) astfel încât* ***M'(p)≥M(p)*** *pentru fiecare poziție p a rețelei.*

Persistență

O rețea Petri (N, M_0) se spune că este ***persistentă (persistent)****, dacă pentru oricare două tranziții validate, executarea uneia dintre ele nu o invalidează pe cea de a doua.*

Așadar într-o rețea persistentă, o tranziție, odată ce a fost validată, rămâne validată până la executarea ei.

Rețeaua Petri din fig. 4.5 (b) este persistentă.

Distanță sincronă

Fie t_1 și t_2 două tranziții ale rețelei (N,M_0). Se numește ***distanță sincronă*** *între t_1 și t_2 numărul (finit sau nu):*

$d(t_1, t_2)=max_\sigma|\overline{\sigma}(t_1)-\overline{\sigma}(t_2)|$,

unde σ notează o secvență de executări pornind din orice marcaj M∈R(M_0), iar $\overline{\sigma}(t_i)$, i=1, 2, notează numărul de executări ale tranziției t_i, i=1, 2, în secvența σ.

Din punct de vedere practic, când rețeaua modelează un proces de tipul condiții-evenimente, distanța sincronă constituie o metrică pentru gradul de dependență mutuală dintre două evenimente.

Exemplu:

Se consideră rețeaua Petri din fig. 4.7. Se constată că pentru evenimentele corespunzătoare tranzițiilor t_2 și t_3 distanța sincronă este $d(t_2, t_3)=2$. Într-adevăr, pentru marcajul atins după executarea lui t_3, secvența σ care poate maximiza modulul prin definiția distanței sincrone este dată de t_2, t_4, t_1, t_2 pentru care $\bar{\sigma}(t_2)=2$ (se execută de două ori) și $\bar{\sigma}(t_3)=0$ (nu se execută niciodată). Evident același rezultat se obține (prin simetrie) dacă se pornește de la marcajul atins după executarea lui t_2.

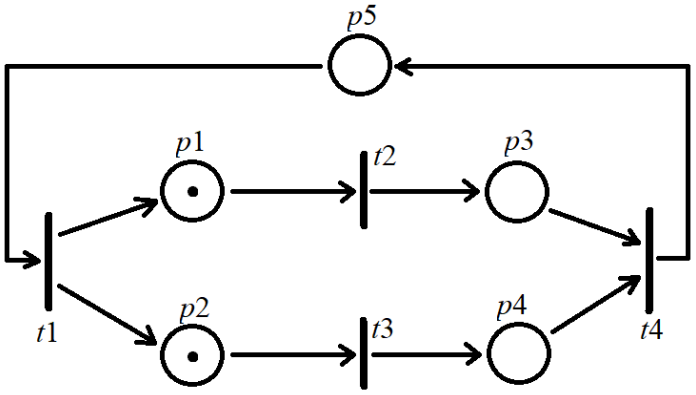


Fig. 4.7.

5. ANALIZA PROPRIETĂȚILOR REȚELELOR PETRI

5.1. Ecuația de stare

Se consideră o rețea Petri pură N (nu există bucle autonome) cu n tranziții și m poziții. Se numește ***matrice de incidență*** (*incidence matrix*) a rețelei, o matrice $\mathbf{A=[a_{iji}]}$ de dimensiuni $n \times m$, ale cărei elemente sunt numere întregi:

$$a_{ij}=a_{ij}^{+}-a_{ij}^{-}; \quad i=1, ..., n, \ j=1, ..., m \tag{1}$$

unde:

- $a_{ij}^{+}=W(t_i, p_j)$ este ponderea arcului *de la* tranziția t_i, *către* poziția sa de ieșire p_j;
- $a_{ij}^{-}=W(p_j, t_i)$ este ponderea arcului *către* tranziția t_i, *de la* poziția sa de intrare p_j.

Matricea $\mathbf{A^{+}=[a_{ij}^{+}]}$ (de dimensiune $n \times m$) este referită drept ***matrice de incidență de ieșire***.
Matricea $\mathbf{A^{-}=[a_{ij}^{-}]}$ (de dimensiune $n \times m$) este referită drept ***matrice de incidență de intrare***.

Din punct de vedere ale aplicațiilor, scrierea matricei de incidență A se poate face global, construind, mai întâi, matricele A^{+} și A^{-}, după care se efectuează diferența:

$$A=A^{+}-A^{-}, \tag{2}$$

care este, evident, echivalentă cu modul de definire (1).

Conform regulii tranziției (simplă), se observă că a_{ij}^{-} și a_{ij}^{+} reprezintă numărul de jetoane îndepărtate și respectiv adăugate în poziția p_j, atunci când tranziția t_i se execută o singură dată.

Se consideră o secvență de executări de tranziții și se presupune că cea de a k-a executare din această secvență are loc în tranziția t_i, adică tranziția desemnată prin t_i se află pe locul k în secvența de executări:

σ =	t_a	t_b	...	t_i	...
	locul 1	locul 2		locul k	

Acest lucru poate fi descris global cu ajutorul unui vector coloană $\boldsymbol{u}_k$ (de dimensiune $n \times 1$) ale cărui elemente sunt toate 0, cu excepția celui de al i-lea element care este 1 (corespunzător tranziției t_i, unde are loc cea de a k-a executare a secvenței).

Forma particulară a lui $\boldsymbol{u}_k$, face ca vectorul coloană $m \times 1$, rezultat din înmulțirea $\boldsymbol{A^T u}_k$ să reprezinte tocmai cea de a i-a coloană a matricei A^T, respectiv cea de a i-a linie a matricei A.

Conform celor discutate mai sus, linia i a lui A (adică a_{ij}, $j=1, ..., m$) are semnificația *schimbării marcajului rețelei* ca urmare a executării tranziției t_i. Deci, revenind la vectorul coloană $\boldsymbol{A^T u}_k$, acesta conține toate schimbările de marcaj rezultate la a k-a executare din secvența considerată (executare ce s-a presupus a avea loc în tranziția t_i).

Pe de altă parte, schimbarea de marcaj în urma celei de-a k-a executări poate fi scrisă drept $M_k - M_{k-1}$, unde M_{k-1}, M_k notează marcajul după cea de-a $(k-1)$-a și respectiv a k-a executare din secvența considerată.

Cum asupra lui k (ce referă executarea din secvență) și a lui i (ce referă tranziția unde are loc această executare) nu au fost puse nici un fel de condiții, rezultă că, în general, schimbarea de marcaj după cea de a k-a executare este descrisă de egalitatea:

$$M_k - M_{k-1} = A^T u_k,\ k=1, 2, \ldots \quad (3)$$

Această egalitate este uzual folosită sub forma echivalentă:

$$\boldsymbol{M_k = M_{k-1} + A^T u_k,} \qquad \boldsymbol{k=1, 2, \ldots} \quad (4)$$

și reprezintă ***ecuația de stare*** (*state equation*) a rețelei Petri.

Vectorul u_k se numește ***vector de executare*** sau **vec*tor de control***.

Exemplu

Se consideră rețeaua Petri din fig. 5.1.

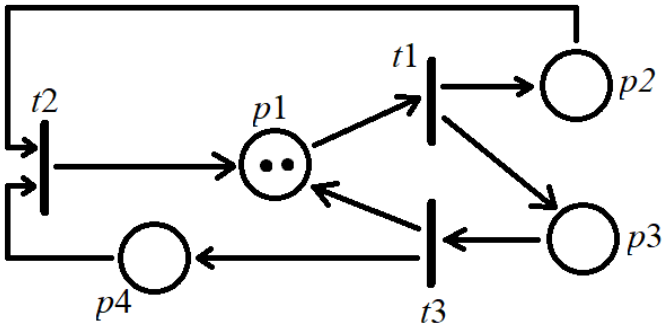


Fig. 5.1.

Matricea de incidență se construiește conform:

$$A = A^+ - A^- = \begin{matrix} \\ t_1 \\ t_2 \\ t_3 \end{matrix} \begin{matrix} \begin{matrix} p_1 & p_2 & p_3 & p_4 \end{matrix} \\ \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 2 \end{bmatrix} \end{matrix} - \begin{matrix} \\ t_1 \\ t_2 \\ t_3 \end{matrix} \begin{matrix} \begin{matrix} p_1 & p_2 & p_3 & p_4 \end{matrix} \\ \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{matrix} = \begin{bmatrix} -2 & 0 & 0 & 0 \\ 0 & -1 & 0 & 2 \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

și permite scrierea imediată a ecuației de stare (4).

Considerând marcajul inițial $M_0=(2\ 0\ 1\ 0)^T$ care validează tranziția t_3, marcajul M_1 rezultat după executarea lui t_3 se obține din ecuația de stare pentru vectorul de control $u_1 = \begin{matrix} t_1 & t_2 & t_3 \end{matrix}\ (\ 0\ \ 0\ \ 1\ )^T$.

Concret, ecuația de stare (4) are forma:

$$\begin{bmatrix} 2 \\ 0 \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} -2 & 1 & 1 \\ 1 & -1 & 0 \\ 1 & 0 & -1 \\ 0 & -2 & 2 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 3 \\ 0 \\ 0 \\ 2 \end{bmatrix}$$

Rezultatul $M_1=(3\ 0\ 0\ 2)^T$ se confirmă imediat și prin aplicarea regulii tranziției în rețeaua Petri din figură.

Utilizând ecuația de stare (4) se pot aborda probleme de *accesibilitate*.

Se presupune că un marcaj de destinație M_d este accesibil M_0 prin secvența de executări u_1, u_2, ..., u_d.

Scriind ecuația de stare (4) pentru k=1, 2, ..., d și sumând, se obține:

$$M_d=M_0+A^T*\sum_{k=1}^{d}u_k, \quad (5)$$

care poate fi rescrisă sub forma:

$$A^T x=\Delta M, \quad (6)$$

unde:

$$x=\sum_{k=1}^{d}u_k, \quad (7)$$

$$\Delta M=M_d-M_0. \quad (8)$$

Vectorul coloană *x*, de dimensiune *n*x1, are toate elementele întregi, nenegative și se numește *vectorul numărului de executări posibile*.

Cel de-al *i*-lea element al vectorului *x*, *i*=1, ..., *n*, conține numărul de executări ale tranziției t_i, *i*=1, ..., *n*, în secvența ce transferă M_0 în M_d.

Condiție de tip necesar pentru accesibilitate

Teorema 1

Dacă marcajul M_d este accesibil din M_0, atunci are loc egalitatea:

$$Rang\ A^T=rang[A^T\ [:]\Delta M], \quad (9)$$

(↑ completat cu coloana)

unde *A* notează matricea de incidență definită prin $A=A^+-A^-$, iar ΔM este diferența de marcaj $\Delta M=M_d-M_0$.

Importanța practică a Teoremei 1 este dată de forma *echivalentă*, care oferă o *condiție suficientă pentru neaccesibilitate*.

Teorema 2

Un marcaj M_d nu este accesibil din M_0, dacă egalitatea de ranguri din (9) nu este satisfăcută.

Observație:
Teorema 1 *nu garantează* atingerea marcajului M_d, pornind din M_0, este doar *necesară* pentru a realiza transferul din M_0 în M_d.

Exemplu:

Se consideră rețeaua Petri din fig. 5.1. În exemplu s-a constatat că executarea tranziției t_3 asigură transferul din $M_0=(2\ 0\ 1\ 0)^T$ în $M_1=(3\ 0\ 0\ 2)^T$. După cum era de așteptat, pentru $\Delta M=M_1-M_0$, condiția necesară formulată de Teorema 1 este satisfăcută, adică:

$$\text{cu: } \underbrace{\begin{bmatrix}3\\0\\0\\2\end{bmatrix}}_{M_1}-\underbrace{\begin{bmatrix}2\\0\\1\\0\end{bmatrix}}_{M_0}=\underbrace{\begin{bmatrix}1\\0\\-1\\2\end{bmatrix}}_{\Delta M} \Rightarrow rg\underbrace{\begin{bmatrix}-2&1&1\\1&-1&0\\1&0&-1\\0&-2&2\end{bmatrix}}_{A^T}=2=rg\underbrace{\begin{bmatrix}-2&1&1&1\\1&-1&0&0\\1&0&-1&-1\\0&-2&2&2\end{bmatrix}}_{A^T\quad\ \underbrace{\ }_{\Delta M}}.$$

În schimb, dacă interesează atingerea marcajului M_2-(3 0 1 0)^T pornind din M_0, se constată că noul $\Delta M=M_2-M_0=(1\ 0\ 0\ 0)^T$ conduce la:

$$rg\begin{bmatrix}-2 & 1 & 1\\ 1 & -1 & 0\\ 1 & 0 & -1\\ 0 & -2 & 2\end{bmatrix} = 2 \neq 3 = rg\begin{bmatrix}-2 & 1 & 1 & 1\\ 1 & -1 & 0 & 0\\ 1 & 0 & -1 & -1\\ 0 & -2 & 2 & 2\end{bmatrix},$$

ceea ce, conform Teoremei 2, înseamnă că M_2 nu este accesibil din M_0.

5.2. Arborele de acoperire (accesibilitate)

Fiind dată o rețea Petri (N, M_0), pornind de la marcajul inițial M_0, modificarea marcajelor ca urmare a executării tranzițiilor poate fi reprezentată sub forma unui arbore, denumit ***arbore de acoperire*** (*coverability tree*). În acest arbore, M_0 este rădăcina, iar marcajele generate sunt noduri; fiecare arc corespunde executării unei tranziții care transformă marcajul asociat nodului de plecare în marcajul asociat nodului de sosire.

În cazul unei rețele Petri *mărginite*, arborele de acoperire se numește ***arbore de accesibilitate*** (*reachability tree*), deoarece poate cuprinde *toate* marcajele accesibile pornind din marcajul inițial M_0 (care sunt în număr finit).

În această situație, arborele de accesibilitate poate fi utilizat pentru studierea *tuturor* proprietăților comportamentale. Un posibil dezavantaj îl constituie numărul mare de noduri ce poate rezulta în arborele de accesibilitate, ca urmare a complexității rețelei Petri studiate.

În cazul rețelelor Petri *nemărginite*, arborele de acoperire va crește la nesfârșit. Pentru a păstra finititudinea reprezentării de tip arbore de acoperire, se introduce un simbol special ω care poate lua valori *oricât de mari*.

Construcția sistematică a arborelui de acoperire a unei rețele (N, M_0) se desfășoară conform următorului algoritm:

Pas 1. Se stabilește M_0 ca rădăcină și se etichetează M_0 ca „*marcaj nou*”.
Pas 2. Atâta timp cât există cel puțin un marcaj etichetat drept „marcaj nou” se efectuează următorii subpași:

Subpas 2.1. Se selectează un „marcaj nou” M;
Subpas 2.2. Dacă M este identic cu un marcaj de pe drumul de la rădăcină la M, atunci marcajul M se etichetează drept „*marcaj vechi*” și se trece la un alt „marcaj nou”;
Subpas 2.3. Dacă pentru M, nici o tranziție nu este validă, atunci M se etichetează ca „marcaj de deadlock”.
Subpas 2.4. Dacă pentru M există tranziții validate, atunci pentru *fiecare* tranziție t validată se efectuează următoarele etape:

Etapa 2.4.1: se obține marcajul M' care rezultă din executarea tranziției t, pornind de la marcajul M:
Etapa 2.4.2: dacă pe drumul de la rădăcină la M există un marcaj M'' astfel încât $M'(p)\geq M''(p)$ pentru orice poziție p și $M'\neq M''$ (în sensul că pentru cel puțin o poziție p inegalitatea $M'(p)\geq M''(p)$ este strictă), atunci $M'(p)$ se înlocuiește cu ω pentru fiecare

poziție p în care avem inegalitatea strictă $M'(p)>M''(p)$ (într-o formulare echivalentă, marcajul M' acoperă marcajul M'').

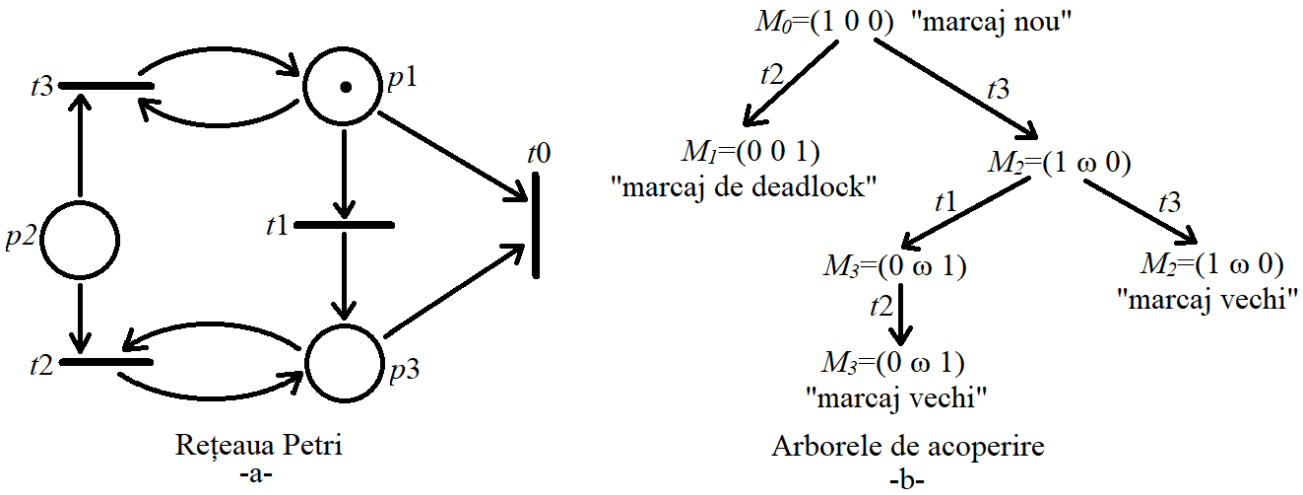
Etapa 2.4.3: se introduce M' ca nod al arborelui de acoperire, se trasează un arc de la M la M' corespunzând tranziției t și se etichetează M' drept „marcaj nou".

Exemplu:

Se consideră rețeaua din fig. 5.2 (a). Aplicarea algoritmului prezentat anterior conduce la arborele de acoperire din fig. 5.2 (b).

Prin executarea tranziției t_1 pornind de la marcajul inițial M_0=(1 0 0), se ajunge la marcajul M_1=(0 0 1), care este un „marcaj de deadlock", întrucât nici o tranziție nu mai este validată pentru M_1.

Executarea tranziției t_3 pentru M_0, conduce la M'=(1 1 0), care acoperă marcajul M_0=(1 0 0) în sensul de la etapa 2.4.2 a algoritmului. Astfel „marcajul nou" este M_2=(1 ω 0), pentru care tranzițiile t_1 și t_3 sunt validate.



Rețeaua Petri
-a-

Arborele de acoperire
-b-

Fig. 5.2.

Executarea lui t_1 transformă M_2 în M_3=(0 ω 1), care este un „marcaj nou".

Executarea lui t_3 pornind din M_2 conduce la un „marcaj vechi", și anume M_2=(1 ω 0).

În fine, pornind din M_3, singura tranziție care se poate executa este t_2, conducând la un „marcaj vechi", și anume M_3=(0 ω 1).

Arborele de acoperire al unei rețele Petri (N, M_0) poate fi folosit pentru studierea proprietăților comportamentale, după cum urmează:

1. Rețeaua este *mărginită* (ceea ce implică faptul că mulțimea de accesibilitate $R(M_0)$ este finită) dacă și numai dacă simbolul ω nu apare în nici un nod al arborelui de acoperire.
2. Rețeaua este *sigură* dacă și numai dacă marcajele din toate nodurile arborelui de acoperire conțin numai 0 și 1.
3. O tranziție t este *viabilă* L_0 (blocată) dacă și numai dacă nu este asociată nici unui arc al arborelui de acoperire.
4. Dacă marcajul M este *accesibil* din M_0, atunci, în arborele de acoperire există un nod ce corespunde unui marcaj M' satisfăcând inegalitatea $M'\geq M$ (în sensul inegalității pe fiecare componentă a marcajului, cazul limită fiind $M'=M$).

În cazul rețelelor *nemărginite*, studierea proprietăților de *viabilitate* cu ajutorul arborelui de acoperire nu este întotdeauna posibilă. Acest fapt se datorează pierderii (prin utilizarea simbolului) a

unor informații numerice concrete, referitoare la marcajul pozițiilor nemărginite (de exemplu creșterea sau descreșterea marcajului).

În cazul unei rețele Petri *mărginite*, arborele de acoperire se numește *arbore de accesibilitate* (*reachability tree*), deoarece poate cuprinde toate marcajele accesibile. În această situație, arborele de accesibilitate poate fi utilizat pentru studierea *tuturor* proprietăților comportamentale. Un posibil dezavantaj îl constituie numărul mare de noduri ce poate rezulta în arborele de accesibilitate, ca urmare a complexității rețelei Petri studiate.

Graful de acoperire (accesibilitate)

Strâns legat de noțiunile de *arbore* de acoperire si de accesibilitate, se introduc *grafurile* de acoperire și, respectiv, de accesibilitate.

Graful de acoperire este asociat unei rețele Petri (N, M_0) este un graf orientat $G=(V, E)$.

Mulțimea nodurilor V este dată de mulțimea tuturor marcajelor *distincte* din arborele acoperire.

Mulțimea arcelor orientate E servește pentru a uni oricare două marcaje M_i, M_j din V, dacă există o tranziție t_k a cărei executare duce de la M_i la M_j; arcele din E corespund arcelor din arborele de acoperire.

Exemplu:

Se consideră rețeaua Petri din fig. 5.2 (a) pentru care arborele de acoperire a fost prezentat în fig. 5.2 (b) (vezi exemplul precedent). Graful de acoperire construit în spiritul definiției de mai sus, este dat în fig. 5.3.

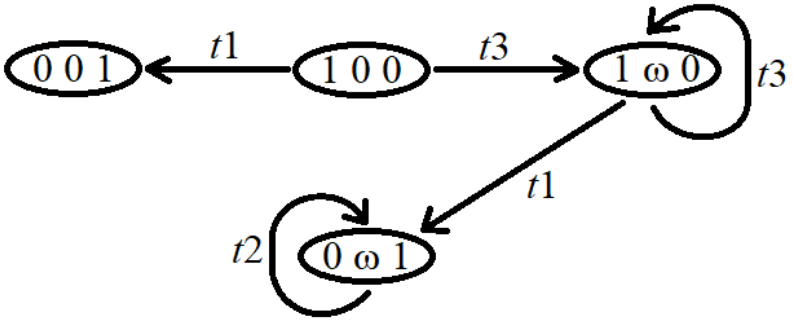


Fig. 5.3 Graful de acoperire pentru rețeaua din fig. 5.2.