1. CONCEPTUL DE SISTEME CU EVENIMENTE DISCRETE

1.1. Considerații generale

Sistemele cu evenimente discrete constituie o clasă de sisteme dinamice neliniare a căror investigație necesită instrumente matematice proprii, diferite de ecuațiile diferențiale sau de ecuațiile cu diferențe, utilizate curent în teoria si practica reglării automate.

La fel ca și în studierea altor tipuri de sisteme, sintagma de "sistem" va fi utilizată pentru a desemna atât o entitate matematică (un model), cât și un obiect fizic (proces), sensul atribuit în fiecare caz rezultând din context.

În funcție de contextul exprimării, prin sistem cu evenimente discrete se înțelege fie un sistem real, fie un model matematic (ce descrie funcționarea unui sistem real), a căror evoluție este raportată la apariția unor evenimente. Astfel, producerea evenimentelor joacă rolul de *cauză* pentru dinamica sistemului și are drept *efect* modificare stărilor sistemului, evidențiind o similitudine cu așa-numita „tratare pe stare" a sistemelor continue sau discrete în timp. Mai mult și în cazul unui sistem cu evenimente discrete se poate vorbi despre o *funcție de tranziție a stărilor,* care formalizează faptul că sistemul trece dintr-o stare în alta numai ca urmare a producerii unui eveniment și că sistemul păstrează starea în care se află până la producerea unui nou eveniment.

Există deosebiri privind interpretarea cauzală a comportării. Dacă în cazul sistemelor clasice, cauzele și efectele sunt valorile unor semnale, care, cel puțin sub raport teoretic, prin variații acoperă intervale (adică mulțimi $\in \Re$), în cazul sistemelor cu evenimente discrete, mulțimea evenimentelor ce pot apărea, precum și mulțimea stărilor în care poate tranzita sistemul sunt discrete (mulțimi $\in$ N).

Dacă dinamica sistemelor clasice este raportată la un *ceas sincron* ce măsoară scurgerea uniformă a timpului (continuu, sau discret – eșantionat cu o anumita perioadă), dinamica sistemelor cu evenimente discrete se raportează la un *ceas asincron,* care marchează succesiunea evenimentelor și, nu în mod obligatoriu, momentele de producere a lor.

În consecință, modelele de tip ecuații diferențiale sau cu diferențe ce caracterizează sistemele clasice s-au dovedit neadecvate pentru descrierea dinamicilor pilotate cu evenimente, motiv pentru care a fost necesar să se recurgă la instrumente matematice de largă factură.

Principalele **caracteristici** ale proceselor cu evenimente discrete de timp real sunt:

- Evenimentele se produc la momente discrete de timp și stările au valori discrete
- Procesele sunt conduse de evenimente nedeterministe (capabile de alegeri) printr-un mecanism care nu este modelat de analistul sistemului; nu este formulată nici o trăsătură stocastică, accentul fiind pus în special pe posibilitatea și nu pe probabilitatea realizării evenimentului;
- Procesele au în general o comportare dinamică internă, reacționând și interacționând cu mediul;
- Procesele operează concurent și comunică unul cu altul(de exemplu prin transmiterea mesajelor pe canale).

Abordările SED pot fi grupate luând în considerare principiile fundamentale și implicit direcțiile de cercetare urmate. În acest sens sunt menționate:

- Limbaje naturale;
- Limbaje formale și automate;
- Limbaje de simulare a evenimentelor discrete;
- Automate stohastice temporizate;
- Mașini cu stări finite;
- Rețele Petri și grafuri de evenimente;
- Procese recursive finite;
- Lanțuri Markov;
- Rețele de cozi de așteptare.

Aplicații ale SED se întâlnesc la descrierea, analiza, sinteza, controlul sau verificarea:

- Sistemelor flexibile de fabricație;
- Rețelelor de comunicație;
- Sistemelor de transport;
- Sistemelor logistice.

Un eveniment poate fi identificat cu o acțiune cum ar fi : ”un buton este apăsat”, ”un client a ajuns” sau ”a fost depășită tensiunea de 100V”.

Pornind de la cele prezentate mai sus se pot enunța principalele ***proprietăți*** ale SED [*Cassandras C. G. – 1993 “ Discrete Event Systems: Modeling and Performance Analysis ”*] :

- Un SED este un sistem pilotat de evenimente, tranzițiile stărilor fiind cauzate de apariția evenimentelor;
- Spațiul stărilor pentru SED este un set discret. După introducerea noțiunii de eveniment, se poate enunța definiția SED.

Definiție (Cassandras C. G.) *Un SED este un sistem cu stări discrete condus de evenimente (event drive) în care evoluția stărilor depinde numai de apariția asincronă a evenimentelor total independent de timp.*

Explorarea dinamicii se poate realiza sub raport calitativ (vizând comportarea logică, independentă de timp) și/sau sub raport cantitativ (vizând comportarea dependentă de timp).

1.2. Sisteme dinamice cu stări continue și sisteme dinamice cu stări discrete

În automatica clasică, formalizarea matematică poate fi orientată asupra modelelor cu stări continue, pentru care spaţiul stărilor este o submulțime al lui R^n.

Acest mod de a privi şi înțelege modelarea proceselor nu este unic, întrucât nenumăratele aplicaţii practice pun în evidenţă sisteme pentru care spaţiul stărilor este o mulţime discretă, numărabilă (finite sau nu).

Exemplul 1

Se consideră funcţionarea recipientului din fig. 1 (a), caracterizată prin evoluţia nivelului de fluid *x(t).* Variabila de stare *x* ia valori în mulţimea [0, x_{max}], iar în fig. 1 (b) este reprezentată o posibilă evoluţie (traiectorie) a lui *x*.

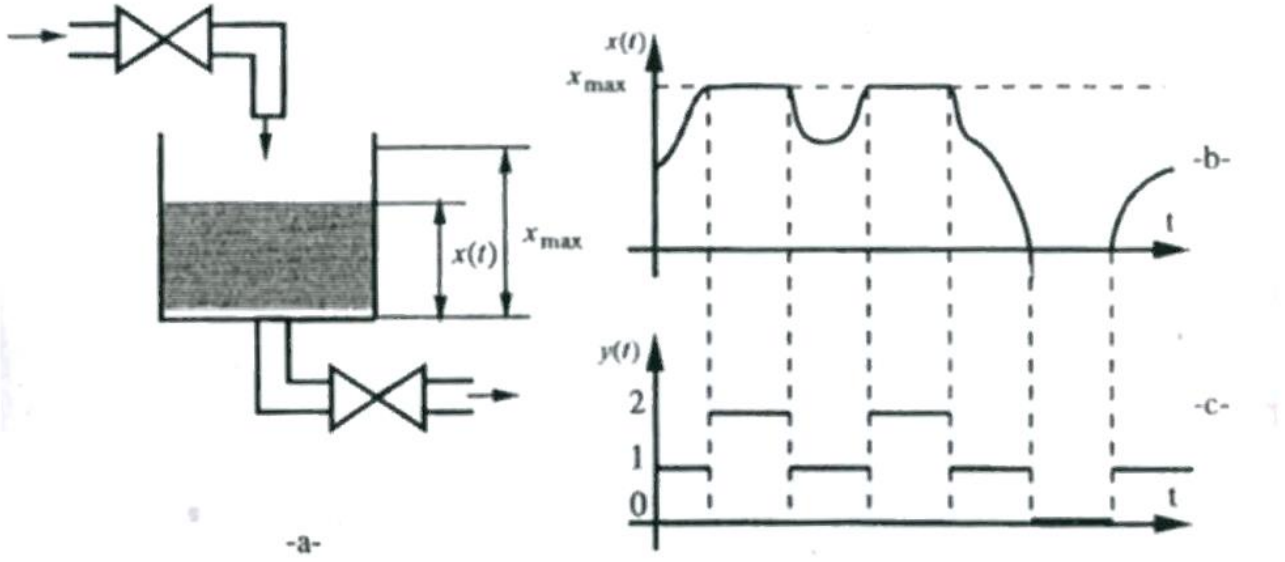


Fig. 1.1 Funcţionarea recipientului din Exemplul 1.

Nu întotdeauna, însă, prezintă interes cunoaşterea efectiv a valorii luate de *x* în intervalul [0, x_{max}]. De pildă, ar putea interesa numai urmatoarele aspecte privind funcţionarea rezervorului:

- dacă recipientul este gol;
- dacă recipientul conţine fluid sub cota x_{max};
- dacă fluidul a atins cota x_{max}.

În acest caz, se poate considera o variabilă de stare discretă $y \in \{0,1,2\}$, unde valorile 0, 1 şi 2 sunt asociate, în ordine, situaţiilor menţionate mai sus. În fig. 1.1 (c) este prezentată evoluţia lui *y(t),* corespunzătoare evoluţiei lui *x(t)* din fig. 1 (b).
Conform diagramei din fig. 1 (c), se constată că tranziţiile dintr-o stare discretă în alta corespund unor evenimente care se produc într-o manieră asincronă:

- tranziţia din 0 în 1 – în rezervorul complet gol tocmai a început să se acumuleze fluid;
- tranziţia din 1 în 0 – rezervorul tocmai s-a golit complet;
- tranziţia din 1 în 2 – fluidul tocmai a atins cota maximă admisibilă x_{max};
- tranziţia din 2 în 1 – nivelul de fluid tocmai a scăzut sub cota maximă admisibilă x_{max}.

1.3. Sisteme dinamice pilotate de timp și sisteme dinamice pilotate de evenimente

Modelele matematice de tip intrare-ieşire sau intrare-stare-ieşire, continue sau discrete în timp, sunt guvernate de o variabilă independentă, cu semnificaţie temporală. Astfel evoluţia acestor modele se raportează la timpul continuu sau discret care se scurge într-o manieră uniformă.
Sistemele cu stări discrete pot pune însă în evidenţă o comportare guvernată de evenimente asincrone, care se petrec la momente de timp repartizate neuniform.

Exemplul 2

Se consideră funcţionarea magaziei automate din fig. 1.2 (a), caracterizată prin evoluţia numărului de piese depozitate x(t).

Se presupune că în magazie se introduce sau se extrage câte o singură piesă şi că durata unei operaţiuni de introducere sau extragere este neglijabilă. Se mai presupune că magazia are o capacitate suficient de mare astfel încât să nu se ajungă la ocuparea completă a spaţiului de depozitare. Să considerăm drept mărimi de intrare funcţiile:

$$u_1(t)=\begin{cases}1; & t \text{ când se introduce o piesă}\\ 0; & \text{în rest}\end{cases}$$

pentru a preciza momentele de introducere a pieselor şi

$$u_2(t)=\begin{cases}1; & t \text{ când se extrage o piesă}\\ 0; & \text{în rest}\end{cases}$$

pentru a preciza momentele de extragere.

Se consideră că introducerea şi extragerea nu pot avea loc simultan, sau : $u_1(t)=1$, şi $u_2(t)=1$ nu există simultan.
Cu aceste ipoteze, o execuţie de stare poate avea forma :

$$x(t^+)=\begin{cases}x(t)+1, dacă: u_1(t)=1; u_2(t)=0\\ x(t)-1, dacă: u_1(t)=0; u_2(t)=1 (şi, x(t)>0)\\ x(t), în\ rest\end{cases},$$

unde t^+>t notează momentul de timp imediat următor lui “t”.

În fig. 1.2 (b) se prezintă evoluţia numărului de piese din magazie (traiectoria lui x(t)) pentru următorul caz:
- momente de introducere ($u_1(t)=1$) :

$t_1 < t_2 < t_3 < t_5 < t_6 < t_{12} < t_{13}$;

- momente de extragere ($u_2(t)=1$):

$t_4 < t_7 < t_8 < t_9 < t_{10} < t_{11}$,

unde $t_k < t_{k+1}$, k=1,…,12.

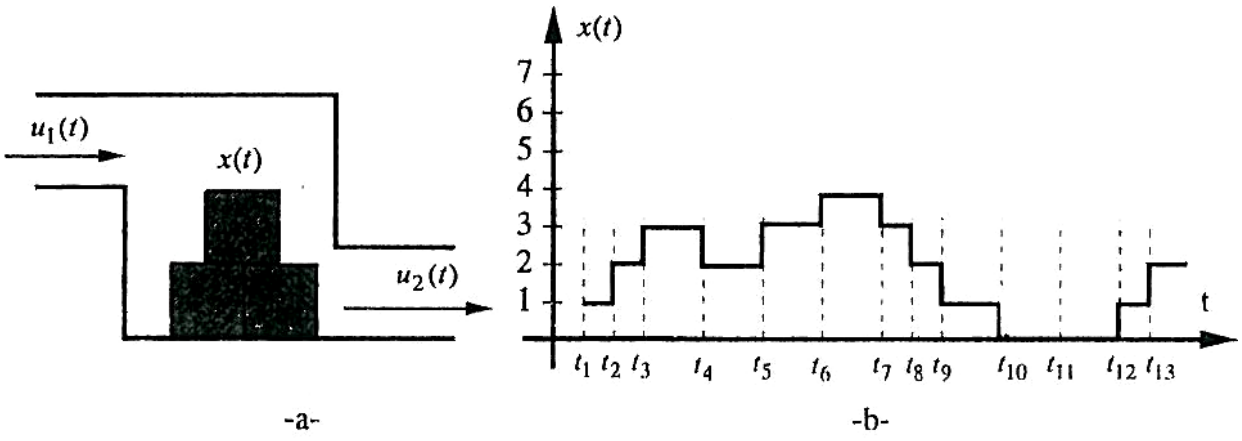


Fig. 1.2 Funcţionarea magaziei automate din exemplul 2.

Se constată că tranziţiile sistemului dintr-o stare în alta sunt dictate către fluxul de evenimente tip introducere şi respective fluxul de evenimente tip extragere. Se observă că alura diagramei din fig. 1.2 (b) rămâne neschimbată dacă valorile numerice t_k, k=1, …,13, se modifică, atâta vreme cât succesiunea acestor monumente rămâne aceeaşi. Mulţimea evenimentelor este o mulţime discretă, numărabilă (finită sau nu).

Conform definiţiei SED, construirea unui model de tip sistem cu evenimente discrete necesită identificarea a două mulţimi discrete (finite sau nu):

X – spaţiul stărilor,

E – mulţimea evenimentelor,

precum și formularea unei descrieri matematice a legităţilor prin care apariţia evenimentelor din mulţimea E determină tranzacţionarea în spaţiul stărilor X .

În spiritul acestei definiții se observă imediat că magazia automată din exemplul 2 nu ar mai constitui un sistem de evenimente discrete în cazul în care introducerea şi extragerea din magazie s-ar realiza cu un ceas sincron a cărui perioadă $T>0$ ar fi cunoscută aprioric. Într-o atare situaţie, am avea de-a face cu un sistem cu stări discrete pilotat de timp, tranziţiile neputând avea loc decât la momentele $t = kT$, $k \in N$.

În realitate, funcţionarea unei magazii automate este condiţionată de ritmul în care se produc piesele (adică intrarea în magazine) şi de cerinţa de distribuţie a pieselor (adică ieşirea din magazine). Întrucât, în general, aceste fenomene nu pot fi privite (în sensul de modelate matematic) ca sincrone cu un anumit ceas de perioadă cunoscută aprioric, necesitatea utilizării, în modelare, a conceptului de sistem cu evenimente discrete apare evidentă.

Evenimentele din mulţimea E pot fi privite ca realizând o structură de ceas asincron (neperiodic) care pilotează desfaşurarea tranziţiilor în X. Această structură de ceas asincron trebuie înţeleasă ca un corespondent al ceasului periodic din cazul sistemelor pilotate de timp. Totuşi, ca şi problematică a comportării, deosebirile sunt fundamentale, întrucât ceasul asincron elimină tocmai condiţia de periodicitate necesară studierii dinamicii sistemelor discrete în timp.

Specificul dinamicii sistemelor cu evenimente discrete necesită un suport matematic adecvat pentru modelare, analiză şi sinteză. Până în prezent nu există o teorie unificată a sistemelor dinamice cu evenimente discrete, modelele şi tehnicile utilizate aflându-şi sorgintea în diferite domenii ale matematicii cum ar fi: teoria reţelelor Petri, teoria automatelor, teoria sistemelor de aşteptare, teoria proceselor Markov, etc.

Sistemele cu evenimente discrete constituie o clasă de sisteme neliniare, după cum reiese din diagrama prezentată în fig 1.3.

Încorporarea sistemelor cu evenimente discrete în diagrama din fig 1.3 este absolut firească, întrucât această clasă de sisteme descrie un anumit tip de dinamică (comportare), care nu este acoperit, la nivel conceptual, de celelalte clase de sisteme cu care se operează curent în diferite domenii ale automaticii. Sistemele cu evenimente discrete constituie o parte integrantă a Teoriei Sistemelor, ca disciplină generală ce oferă suport matematic riguros, necesar modelării și investigării comportamentului proceselor fizico-tehnice.

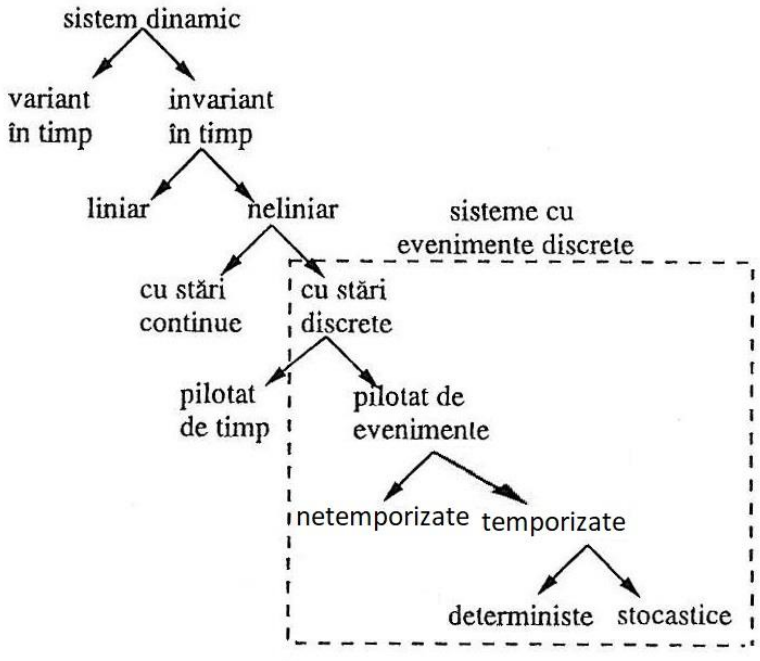


Fig. 1.3 Specificul comportării sistemelor cu evenimente discrete raportat la o clasificare de ansamblu a sistemelor dinamice.

Ceea ce diferă însă de la o clasă de sisteme dinamice la alta este tipul de instrument matematic la care se face apel pentru descrierea cât mai precisă a specificului dinamicii. Astfel este de așteptat faptul că pentru modelarea, analiza și sinteza sistemelor cu evenimente discrete să se utilizeze un suport teoretic propriu studierii acestei clase de siteme.

1.4. Considerații calitative si cantitative in modelarea dinamicii sistemelor cu evenimente discrete. Abstractizarea in studiul SED

În definiţia formulată referitoare la SED nu s-a facut nici o precizare asupra momentelor de timp când apar evenimentele din mulţimea *E*, cu excepţia faptului că producerea lor este asincronă. În cazul în care modelul matematic nu conţine nici o informaţie asupra momentelor de timp când se produc evenimentele $e_1, e_2, \ldots, e_n, \ldots \in E$, avem de a face cu un *model logic* sau *netemporizat*.

Astfel de modele pot pune în evidenţă numai *proprietăţile de tip calitativ* ale comportării sistemelor. Explorarea proprietăţilor calitative se bazează pe tehnici calitative, ce utilizează drept formalism matematic fie *teoria reţelelor Petri*, fie *teoria automatelor*.

În schimb, în cazul când evenimente din E sunt prezentate sub forma unor perechi (e_1, t_1), (e_2, t_2), …, (e_n, t_n) … în care $t_1, t_2, \ldots, t_n, \ldots$ notează momentele de apariţie a evenimentelor respective, avem un *model temporitzat*. Astfel de metode pot pune în evidenţă *proprietaţi de tip cantitativ* ale comportării sistemelor. Cunoaşterea momentelor de timp $t_1, t_2, \ldots, t_n, \ldots$ poate fi de factură deterministă sau stocastică, modelele temporizate divizându-se în *modele deterministe* şi *stocastice*, după cum rezultă şi din diagrama din fig. 1.3. Explorarea proprietăţilor cantitative se bazează pe tehnici cantitative. Iniţial, tehnicile cantitative au făcut apel la aparatul matematic specific teoriei *proceselor Markov* sau *teoriei sistemelor de aşteptare*. Ulterior au fost realizate extinderi ale teoriei reţelelor Petri şi teoriei automatelor, capabile să înglobeze şi aspectul temporal, demonstrându-se totodată că aceste extinderi devin compatibile cu tehnicile cantitative amintite anterior.

Este evident că proprietăţile de natură calitativă posedă un grad mult mai mare de generalitate decât cele cantitative, în sensul că un numar de sisteme ce au o comportare identică sub raport calitativ, pot diferi mult între ele atunci când sunt luate în discuţie şi aspecte cantitative.

Teoria sistemelor cu evenimente discrete poate fi abordată prin intermediul mai multor formalisme. Se va utiliza formalismul reţelelor Petri, întrucât reprezentările de acest tip sunt dublate de un suport grafic cu impact deosebit de eficient asupra înţelegerii la nivel intuitiv a dinamicii sistemelor cu evenimente discrete.

Rețelele Petri pot fi relaționate cu automatele cu stări finite prin faptul că și acestea prezintă în mod explicit funcția tranzițiilor de stare a unui sistem cu evenimente discrete. O rețea Petri este un dispozitiv care manipulează evenimete conform anumitelor reguli, fapt care duce la privirea acesteia ca un automat cu stări finite.

Pentru a evidenția diferența esențială dintre sistemele dinamice continue (SDC) și SED se consideră reprezentările din figura 1.4.

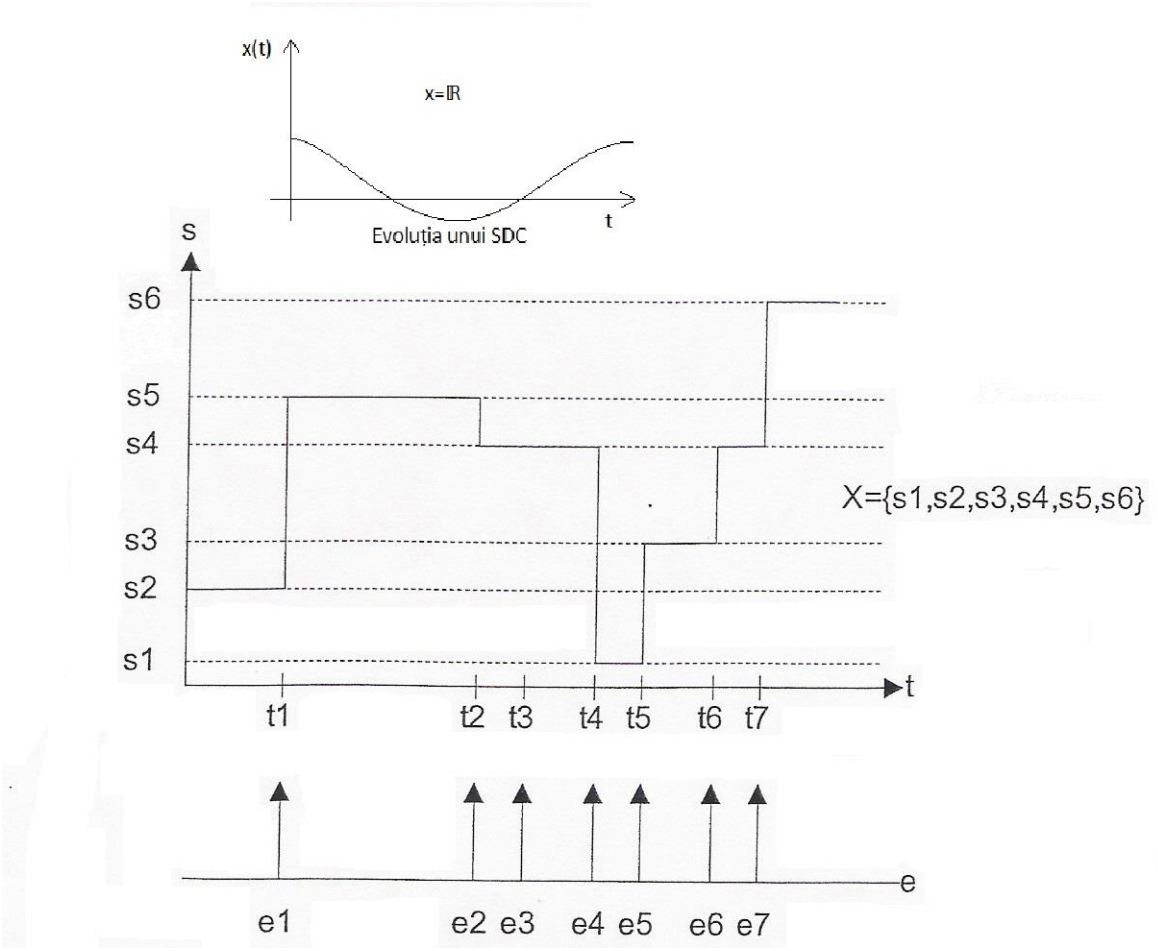


Fig. 1.4 Evoluția unui SED.

- Pentru SDC, spațiul stărilor X este spațiul numerelor reale R, și X poate lua orice valoare din R.

Funcția x(t) este soluția unei ecuații diferențiale de formă general

$x(t)=f(x(t), u(t))$ cu u(t) – intrare.

- Pentru SED – spațiul stărilor este mulțimea discretă

$X=\{S_1, S_2, S_3, S_4, S_5, S_6\}$.

În cadrul acestei secvențe se consideră că evenimentul e_1 apare la momentul de timp $t=t_1$; evenimentul e_2 apare la momentul de timp t_2 și așa mai departe. Conform celor prezentate mai sus starea inițială a sistemului este s_2. Situația considerată este corespunzătoare unui sistem determinist deoarece tranziția dintr-o stare în alta se face după apariția unui eveniment care este unic în sistem.

Secvența de evenimente temporizate corespunzătoare este:

$$(e_1, t_1)\ (e_2, t_2)\ (e_3, t_3)\ (e_4, t_4)\ (e_5, t_5)\ (e_6, t_6)\ (e_7, t_7)\ . \qquad (1)$$

De notat că un eveniment poate avea loc, dar nu este obligatoriu să cauzeze o tranziție a stării- cum este cazul evenimentului e_3 din figură.

Nu există o analogie cu cazul SDC, neexistând o ”ecuație de stare” pentru SED, care să determine ”mecanismul” de apariție a evenimentelor.

Abstractizarea în studiul SED se poate aborda considerând (pornind) de la secvența de evenimente descrisă anterior (1).

Pe baza ei se poate reconstitui evoluția sistemului pornind din orice moment –fig. 1.5.

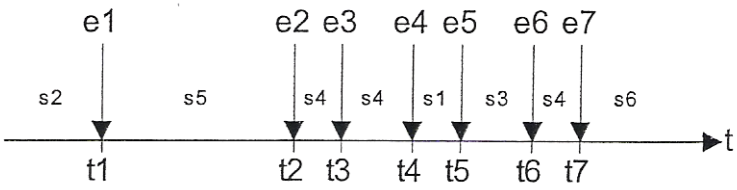


Fig. 1.5 Evoluția stărilor pentru SED-ul considerat.

1.5. Exemple de procese abordate ca SED

1.5.1. Servirea clienţilor într-un sistem de aşteptare

Un sistem de aşteptare se compune din 3 elemente definitorii puse în evidenţă şi de reprezentarea schematizată din fig. 1.6:

- clienţii (entitațile) – care așteaptă în căutarea lor după resurse;
- resursele (serverele);
- firul de așteptare (coada de așteptare).

Caracterizarea completă a sistemului de aşteptare presupune furnizarea următoarelor informaţii:

- numărul de servere;
- capacitatea firului de aşteptare (numărul de clienţi acceptaţi);
- disciplina de servire a clienţilor (în ordinea venirii, după anumiteprioritățţi, aleatorie).

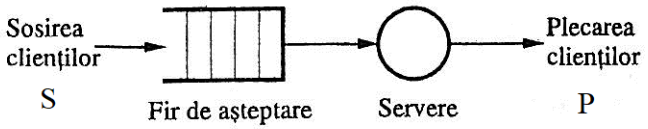


Fig. 1.6 Reprezentarea schematică a unui sistem de aşteptare.

Descrierea ca sistem cu evenimente discrete se bazează pe o mulţime de evenimente *E* ce cuprinde evenimente de două tipuri : *s* (sosire) şi *p* (plecare). Starea sistemului *x(t)* este dată de numărul total de clienţi care se găsesc la un moment arbitrar de timp *t* în sistem (clienţi care aşteaptă pe fir, plus clienţi care sunt serviţi), adică spaţiul stărilor este mulțimea *X = {0, 1, 2, ...}*.

Exemple de clienți:

- Oamenii așteptând într-o stație de autobus sau într-o bancă;
- Mesaje transmise într-un mediu de comunicație;
- Joburi, taskuri sau tranzacții executate într-un calculator;
- Mașinile într-o rețea de drumuri.

Exemple de servere (resurse):

- Funcționarul de la bancă;
- Canalele de comunicație destinate transmiterii de mesaje;
- Procesorul calculatorului sau echipamentului periferic;
- Diferite tipuri de mașini utilizate în procesul de fabricație;
- Semafoarele din intersecții.

Sistemul de aşteptare trebuie privit drept o structură (configuraţie) tipică ce poate fi întâlnită în funcţionarea multor procese fizico-tehnice, ca de pildă:

- aşteptarea joburilor sau taskurilor într-un sistem de calcul pentru a beneficia de o anumită resursă (procesor sau periferic);
- aşteptarea pieselor într-un sistem de fabricaţie pentru a fi prelucrate pe o anumită maşină;
- aşteptarea mesajelor într-un sistem de comunicaţii pentru a fi trasnsmise.

Două sau mai multe sisteme de asteptare pot fi interconectate, dând naştere la o reţea de aşteptare.

1.5.2. Procesarea joburilor într-un sistem de calcul

Funcţionarea sistemelor de calcul se bazează pe organizarea firelor de aşteptare pentru diferite resurse, cum ar fi procesoarele, perifericele etc. Clienţii sunt joburile sau taskurile prezentate în sistemul de calcul. De exemplu, în fig. 1.7 se prezintă schematizat, modul de prelucrare a unui lot de joburi de către un sistem de calcul, în care orice job solicită accesul la UC şi, eventual, la discurile *D1*, *D2*. Modelarea ca sistem cu evenimente discrete se bazează pe identificarea următoarelor tipuri de evenimente ce pot tranziţiona sistemul dintr-o stare în alta:

s - sosirea unui job în lotul de intrare (din universal exterior);

p - plecarea unui job din UC (către universal exterior);

r_1 - rutarea unui job pentru utilizarea discului D1;

r_2 - rutarea unui job pentru utilizarea discului D2;

d_1 - rutarea unui job de la D1 către UC;

d_2 - rutarea unui job de la D2 către UC.

Vectorul de stare este un triplet de forma $(x_{UC}, x_1, x_2)^T$, unde fiecare componentă este un număr natural, limitat superior de capacitatea fiecăruia din sistemele de aşteptare ce intră în structura reprezentată în fig. 1.7. În consecinţă, spaţiul stărilor este o mulţime de forma $X \{(x_{UC}, x_1, x_2)^T\}$. În funcţie de situaţia reală pe care o modelăm, capacitatea sistemelor de aşteptare va fi precizată în clar atunci când este posibil. Dacă se consideră că, în orice condiţii de funcţionare, această capacitate nu poate fi atinsă, variabilele de stare x_{UC}, x_1, x_2 iau valori în mulţimea {0, 1, 2, …, n, …}, fără a se impune mărginirea superioară.

Setul de evenimente, pentru exemplul din figură, constă în sosirile și plecarile la diverse servere, adică:

$$E=\{s, p, r_1, r_2, d_1, d_2\}.$$

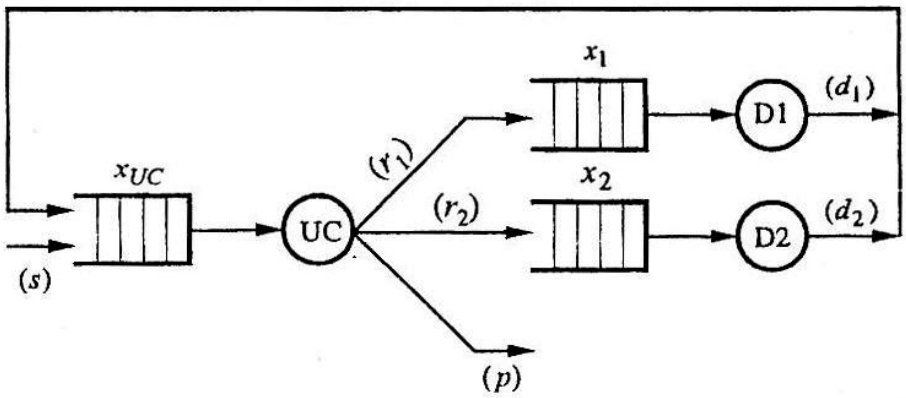


Fig. 1.7 Reprezentarea schematizată a prelucrării unui lot de joburi într-un system.

1.5.3 Prelucrarea pieselor într-un sistem de fabricaţie

În funcţionarea sistemelor de fabricaţie, se utilizează frecvent şiruri de aşteptare sub forma unor depozite (buffer) care permit stocarea pieselor ce trebuie prelucrate pe o anumită maşină. În fig. 1.7 se reprezintă schematic configuraţia unui sistem de fabricaţie alcătuit din două maşini *(M1)* şi *(M2) (server)*, ambele prevăzute cu depozite de intrare. Se presupune că depozitul ce precede *M1* are capacitate nelimitată (practic, nu se poate ajunge la situaţia umplerii complete), în timp ce în depozitul ce precede *M2* se pot stoca numai două piese.

Mulţimea evenimentelor *E* se compune din evenimente de tipul:

s - sosirea unei piese brute în sistemul de fabricaţie;

p - plecarea unei piese prelucrate complet din sistemul de fabricaţie;

c_1 - terminarea prelucrării unei piese pe maşina *M1*.

Setul de evenimente pentru acest exemplu devine $E=\{s, c_1, p\}$.

Pentru a identifica variabilele de stare, pornim de la observaţia că fiecare din cele două maşini reprezintă serverul unei structuri de tipul sistem de aşteptare. În consecinţă, procesul de prelucrare a pieselor poate fi privit ca o conexiune a celor două sisteme de aşteptare, ale căror stări se descriu în conformitate cu cele discutate în paragraful 1.5.1. Astfel vectorul de stare al modelului întregului sistem de fabricaţie are forma $(x_1, x_2)^T$, unde $x_1 \in \{0, 1, 2, \ldots\}$, iar $x_2 \in \{0, 1, 2, 3\}$, adică spaţiul stărilor este mulţimea:

$$X_1=\{(x_1, x_2)^T / x_1 >=0, x_2 \in \{0, 1, 2, 3\}\}.$$

Examinând cu atenţie funcţionarea sistemului de fabricaţie, se observă că modelarea bazată pe X_1 definit mai sus, nu poate surprinde următorul aspect privind utilizarea maşinii *M1*. În cazul în care *M2* prelucrează o piesă şi alte două piese aşteaptă în depozitul ce precede M2, piesa de pe maşina *M1* poate fi în curs de prelucrare, sau prelucrată complet, dar ramasă pe maşină din lipsă de spaţiu pentru depozitarea ei, situaţie care este cunoscută sub numele de blocaj. Astfel, dacă prin variabilele de stare x_1 şi x_2, identificăm stările celor două maşini (fără să ţinem cont de numărul de piese din fiecare sistem de aşteptare), putem utiliza drept spaţiu al stărilor mulţimea:

$$X_2 = \{ (x_1, x_2) / x_1 \in \{N, L, B\}, x_2 \in \{N, L\} \},$$

unde simbolurile N, L, B notează următoarele stări posibile:

N - maşina este neutilizată (Idle),

L - maşina lucrează,

B - maşina este blocată.

Se observă că prin utilizarea spaţiului stărilor X_2, modelul poate face distincţie între starea *B* sau *L* a maşinii *M1*, pentru situaţia semnalată anterior. Totuşi, de data aceasta, nu mai avem informaţii privind numărul de piese din fiecare sistem de aşteptare. Putem încorpora şi aceste informaţii în model, revenind la spaţiul stărilor X_1 şi observând că blocajul lui *M1* nu poate apărea decât pentru $x_2 = 3$. Cu alte cuvinte, $x_2 = 3$ poate corespunde la două stări distincte ale sistemului de fabricaţie (care diferă prin situaţia în care se află *M1*):

- $x_2 = 3_L$ - maşina M1 lucrează ;
- $x_2 = 3_B$ - maşina M1 este blocată.

Astfel spaţiul stărilor definit prin mulţimea :

$$X_3 = \{(x_1, x_2)^T \ / \ x_1 > 0,\ x_2 \in \{0, 1, 2, 3_L, 3_B\}\}.$$

Permite includerea în model atât a stărilor individuale a maşinilor, cât şi a numărului de piese aflate în fiecare din sistemele de aşteptare.

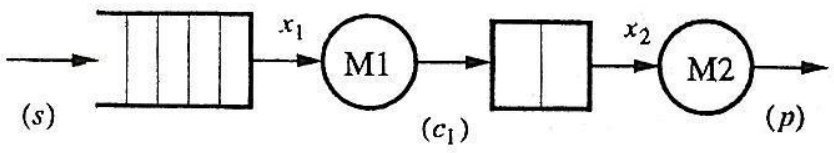


Fig. 1.8 Reprezentarea schematizată a prelucrării unui lot de piese într-un sistem de fabricaţie.

2. REȚELE PETRI

Teoria rețelelor Petri (Petri nets) își are originea în teza de doctorat a lui **Carl Adam Petri "Kommunication mit Automaten"**. Aceasta a fost susținută în anul 1962 la Universitatea Tehnică din Darmstadt.

Utilizate inițial în studiul și analiza fluxului informației, rețelele Petri s-au dovedit a fi deosebit de utile în descrierea operațiilor paralele, concurente, asincrone, distribuite, posibil nedistribuite. Ele pot fi folosite la reprezentarea interconectării dintre entități, la analiza sau sinteza sistemelor cu evenimente discrete în general, dar prin extensie și la sisteme continue.

Utilizarea modelării, sintezei și analizei prin intermediul rețelelor Petri, prescurtat RP, a dus la rezultate semnificative în următoarele domenii:

- Sisteme software distribuite;
- Sisteme de operare și compilare;
- Baze de date distribuite;
- Limbaje formale;
- Programe logice;
- Programe paralele și concurente;
- Sisteme de control pentru procese industriale;
- Sisteme flexibile de fabricație;
- Sisteme cu evenimente discrete;
- Sisteme multiprocesor;
- Automate programabile;
- Circuite și structuri asincrone;
- Rețele locale;
- Filtre digitale;
- Modele de decizie.

Rețelele Petri sunt instrumente teoretice bazate pe grafuri. Reprezentarea grafică a rețelelor Petri facilitează înțelegerea și utilizarea lor. De asemenea, li se poate atașa un formalism matematic util pentru analiză.

Construirea modelelor netemporizate stă la baza analizei și sintezei sistemelor dinamice cu evenimente discrete și are drept obiectiv descrierea de tip calitativ (logic) a dinamicii acestora. Formalismul rețelelor Petri netemporizate oferă unul dintre cele mai performante instrumente utilizate în acest scop.

2.1. Reţele Petri netemporizate

O *reţea Petri* se compune dintr-un tip particular de *graf orientat* notat *N* și o *stare inițială* M_0 sau X_0, denumită *marcaj iniţial* (se va nota cu M_0 sau cu X_0). Graful *N* al rețelei Petri este orientat, ponderat şi bipartit, constând din două tipuri de noduri, denumite *poziţii* sau *locaţii*, notate "*p*" şi

respectiv *tranziții* notate cu ”*t*”; *arcele* pleacă fie de la o poziție la o tranziție, fie de la o tranziție la o poziție (nu există arce care să conecteze două poziții între ele, sau două tranziții între ele!).
Ca simbolizare grafică, pozițiile se reprezintă prin cercuri, iar tranzițiile prin bare sau dreptunghiuri (fig. 2.1).

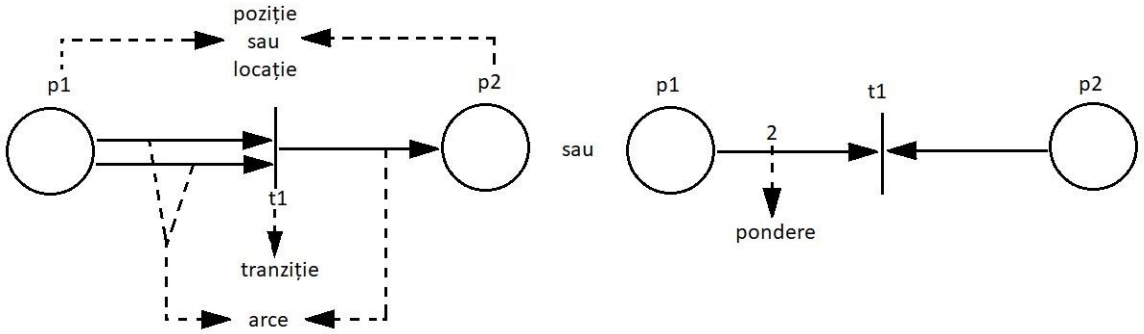


Fig. 2.1.

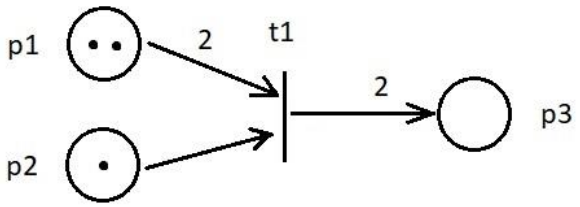
Arcele sunt etichetate cu ponderile lor (valori întregi, pozitive); un arc cu ponderea *k* poate fi privit ca o mulțime de *k* arce paralele cu o pondere unitară. Se formalizează prin funcția de ponderare:

$w(p_i, t_j)=k$ sau $w(t_j, p_i)=k$.

Etichetele pentru pondere unitară se omit în reprezentările grafice uzuale.

Un *marcaj* sau o *stare* atribuie fiecărei poziţii un număr întreg mai mare sau egal cu 0. Dacă un marcaj atribuie poziţiei *p* întregul $k \geq 0$, se spune că *p* este marcat cu *k jetoane*. Din punct de vedere grafic, în cercul corespunzător poziţiei *p* se vor plasa *k* buline(puncte).

Orice marcaj M (sau X) este un *vector m*-dimensional, unde *m* notează numărul total al poziţiilor; componenta *p* a lui *M* (sau *X*) notată *M(p)* (sau *X(p)*) semnifică numărul de jetoane din poziţia *p*.



$$M_0(p_1) = 2;\ M_0(p_2) = 1;\ M_0(p_3) = 0;$$
$$X_0 \qquad X_0 \qquad X_0$$

Fig 2.2.

În problemele de modelare ce utilizează conceptele de *condiţii* şi *evenimente*, poziţiile reprezintă condiţii şi tranziţiile reprezintă evenimente. O tranziţie (eveniment) posedă un număr de poziţii de intrare şi ieşire, care reprezintă *pre-condiţii* şi respectiv *post-condiţii* pentru evenimentul în cauză. Prezenţa unui jeton într-o poziţie trebuie înţeleasă ca valoare logică *adevărat* pentru condiţia asociată respectivei poziţii.

Aspectele prezentate anterior se formalizează matematic prin următoarea *definiţie*:

Definiție: ***Rețea Petri netemporizată***

Un graf (sau o structură) de rețea Petri este un graf bipartit ponderat, formalizat matematic prin cvintuplul

$PN=(P, T, F, W, M_0/X_0)$ (topologia rețelei) unde:

- P reprezintă mulţimea finită de poziţii, unde $P=\{p_1, p_2,..., p_m\}$;
- T reprezintă mulţimea finită de tranziţii, unde $T=\{t_1, t_2, ..., t_n\}$;
- $F \subseteq (P \times T) \cup (T \times P)$ este mulţimea arcelor de la poziții la tranziții și de la tranziții la poziții, fiecare arc fiind reprezentat prin (p_i, t_j), respectiv (t_j, p_i), unde $i, j \in N$;
- $W:F \rightarrow \{1,2,3,...\}$ este funcția de ponderare a arcelor;
- M_0(sau X_0)$:P \rightarrow \{0,1,2,3...\}$ este funcţia de marcaj iniţial.

Observaţii!

- mulţimile *P* şi *T* sunt disjuncte $P \cap T=\{\phi\}$.
- pentru a asigura obiectul definiţiei de mai sus, mulţimile *P* şi *T* satisfac condiţia $P \cup T \neq \{\phi\}$.
- o structură de reţea Petri *N=(P, T, F, W)*, fără nici o specificaţie referitoare la marcaj, se va nota cu *N*, notaţie care desemnează *topologia reţelei*.
- o reţea Petri cu marcaj iniţial M_0 se va nota prin (N, M_0).
- o rețea Petri cu marcaj oarecare *M* se va nota prin *(N, M)*.

În descrierea unui graf de rețea Petri, se folosește $I(t_j)$ pentru a reprezenta mulțimea pozițiilor de intrare pentru tranziția t_j și $O(t_j)$ pentru a reprezenta mulțimea pozițiilor de ieșire pentru tranziția t_j.

$I(t_j) = \{ p_i \in P: (p_i, t_j) \in F; i, j \in N\}$; $O(t_j) = \{p_i \in P: (t_j, p_i) \in F; i, j \in N \}$.

Pentru a descrie tranzițiile de intrare pentru o poziție p_i se folosește $I(p_i)$, iar pentru a descrie tranzițiile de ieșire se folosește $O(p_i)$.

Exemplu de graf de rețea Petri simplă:

Se consideră rețeaua Petri din fig. 2.1 definită prin:

$P=\{ p_1, p_2\}$, $T=\{t_1\}$, $F=\{(p_1, t_1), (t_1, p_2)\}$;

$w(p_1, t_1)=2$; $w(t1, p2)=1$.

În acest caz, $I(t_1)=\{p_1\}$ și $O(t_1)=\{p_2\}$.

Exemplu

Se consideră rețeaua Petri din figura 3:

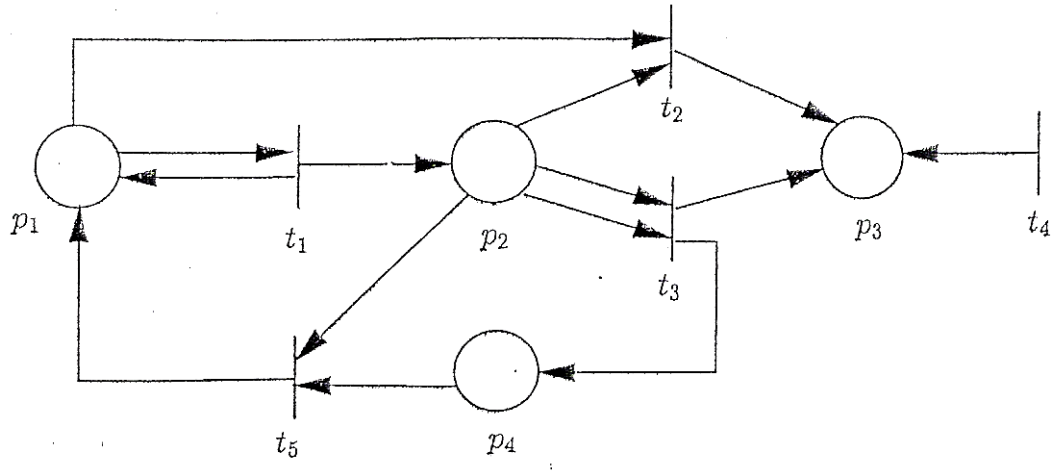


Fig. 2.3.

Rețeaua Petri reprezentată este specificată prin:

$P = \{p_1, p_2, p_3, p_4\}$ $T = \{t_1, t_2, t_3, t_4, t_5\}$

$F = \{(p_1, t_1), (p_1, t_2), (p_2, t_2), (p_2, t_3), (p_2, t_5), (p_4, t_5), (t_1, p_1), (t_1, p_2), (t_2, p_3), (t_3, p_3), (t_3, p_3), (t_4, p_3), (t_5, p_1)\}$

$w(p_1, t_1)=1$; $w(p_1, t_2)=1$; $w(p_2, t_2)=1$; $w(p_2, t_3)=2$;
$w(p_2, t_5)=1$; $w(p_4, t_5)=1$; $w(t_1, p_1)=1$; $w(t_1, p_2)=1$;
$w(t_2, p_3)=1$; $w(t_3, p_3)=1$; $w(t_3, p_4)=1$; $w(t_4, p_3)=1$;
$w(t_5, p_1)=1$.

Se observă că tranziția 4, t_4 – nu are o poziție de intrare; evenimentul corespunzător lui t_4 are loc necondiționat. În contrast, evenimentul corespunzător tranziției t_2 de exmplu, depinde de anumite condiții relative la pozițiile p_1 și p_2.

Marcajul unei rețele Petri și vectorii de stare

Asocierea de jetoane pozițiilor unui graf determină marcarea acestuia.
În urma marcării pozițiilor unei rețele Petri putem vorbi de *starea rețelei.*
Starea rețelei Petri se definește ca fiind vectorul marcajelor.

Marcajul M (sau X) definește vectorul linie:

$$M = M[M(p_1), M(p_2), \ldots, M(p_n)]$$

sau

$$X = [X(p_1), X(p_2), \ldots, X(p_n)]$$

cu n, numărul de poziții din graf.

Numărul de jetoane corespunzător unei poziții p_i îl regăsim în poziția *i* a vectorului M sau X.

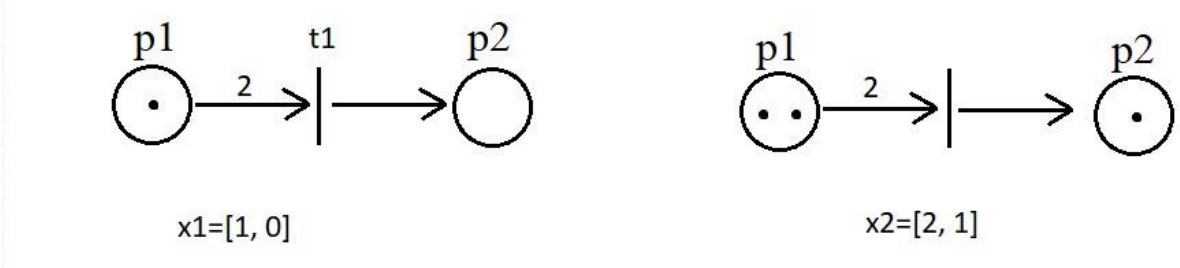
În urma marcării, o rețea Petri va deveni o rețea marcată și va fi simbolizată prin cvintuplul (P, T, F, W, X).

Exemplu:

Două posibile marcări definite de vectorii linie:

$x_1 = [1, 0]$ și $x_2 = [2, 1]$

sunt reprezentate în figură:



2.2. Validarea şi executarea tranziţiilor în rețele cu capacitate infinită

Marcajul unei reţele Petri are semnificaţie de *stare a reţelei* şi se poate modifica în conformitate cu următorul procedeu denumit *regula tranziţiei* (validare şi executare) sau *regula simplă a tranziției*.

Consideraţii:

a) Se spune că o tranziţie *t* este validate (*enabled*) dacă fiecare poziţie de intrare (predecesor) *p* a lui *t* este marcată cu cel puţin *w*(*p*, *t*) jetoane, unde *w*(*p*, *t*) notează ponderea arcului de la *p* la *t*.

b) O tranziţie validată poate sau nu sa fie *executată* sau *declanşată* (*fired*), după cum evenimentul asociat tranziţiei are sau nu loc.

c) Executarea unei tranziţii validate îndepărtează *w*(*p*, *t*) jetoane din fiecare poziţie de intrare (*predecesor*) *p* a lui *t* şi adaugă *w*(*t*, *p*) jetoane la fiecare poziţie de ieşire (*successor*) *p* a lui *t*, unde *w*(*t*, *p*) este ponderea arcului de la *t* la *p*.

Regula (simplă) a tranziției

O tranziție $t_j \in T$ a unei rețele Petri se numește validată dacă

$$x(p_i) \geq w(p_i, t_j), \qquad \text{oricare ar fi } p_i \in I(t_j).$$

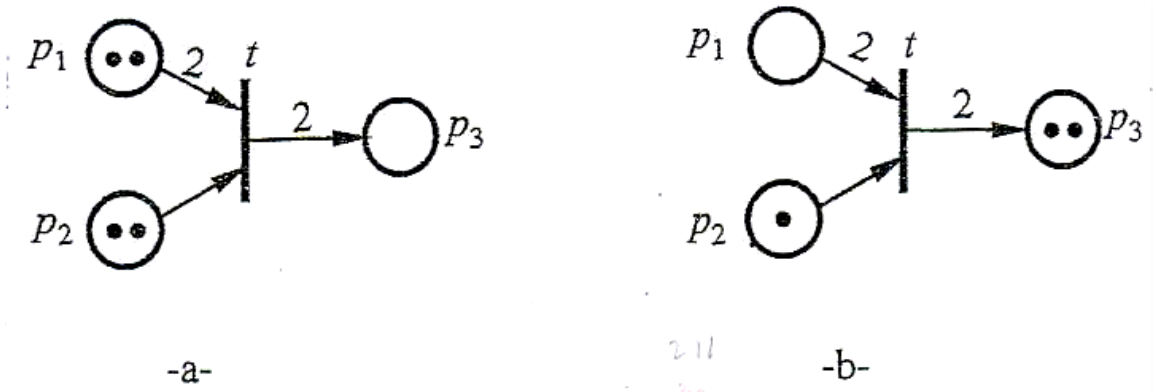
Cu alte cuvinte, tranziția t_j din rețeaua Petri este validată dacă numărul jetoanelor din poziția sa de intrare p_i este mai mare sau egal cu ponderea arcului respectiv.

Exemplul 2.1.1

Se consideră reţeaua Petri definită prin:

$P=\{p_1, p_2, p_3\}$, $T=\{t\}$;
$F=\{(p_1, t), (p_2, t)\}\cup\{(t, p_3)\}$;
$w(p_1, t)=2$, $w(p_2, t)=1$, $w(t, p_3)=2$;
$M_0(p_1)=2$, $M_0(p_2)=2$, $M_0(p_3)=0$.

Reprezentarea grafică a reţelei este dată în fig. 2.4 (a).



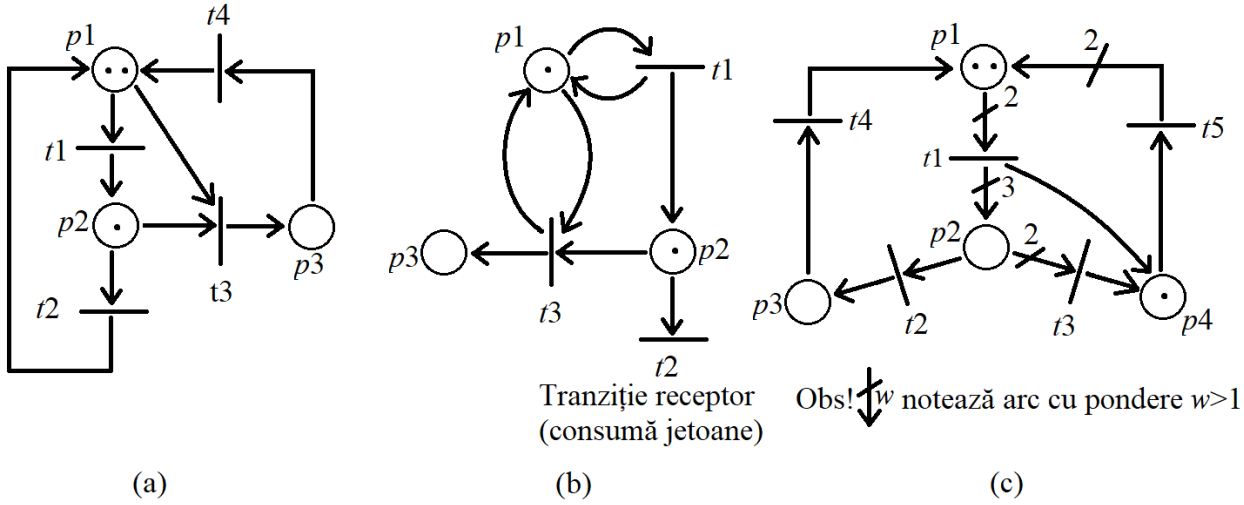
-a- -b-

Marcajul înainte de executarea lui *t* Marcajul după executarea lui *t*

Fig. 2.4.

Tranziția *t* este validată și executarea ei conduce la marcajul $M(p_1)=0$; $M(p_2)=1$, $M(p_3)=2$, reprezentat în fig. 2.4 (b).

Exemple:



(a) (b) (c)

Fig. 2.5.

Fie rețelele Petri cu topologia și marcajul inițial indicat în fig. 2.5 (a), (b), (c).

Să se precizeze care dintre tranziții sunt validate și marcajul care rezultă după executarea fiecăreia dintre tranziții.

Soluție:

1. Aplicând regula simplă a tranziției rețelei din fig. (a) cu marcajul inițial $M_0 = (2, 1, 0)$ se obține:
   - tranziția t_1 este validată deoarece $M_0(p_1) > w(p_1, t_1)$; prin executarea lui t_1 se ajunge din M_0 la marcajul $M_1 = (1, 2, 0)$;
   - tranziția t_2 este validată deoarece $M_0(p_2) = w(p_2, t_2)$; prin executarea lui t_2 se ajunge din M_0 la marcajul $M_2 = (3, 0, 0)$;
   - tranziția t_3 este validată deoarece $M_0(p_1) > w(p_1, t_3)$ și $M_0(p_2) = w(p_2, t_3)$; prin executarea lui t_3 se ajunge din M_0 la marcajul $M_3 = (1, 0, 1)$.

2. Analog pentru rețeaua din fig. (b) cu marcajul inițial $M_0 = (1, 1, 0)$ rezultă:
   - tranziția t_1 este validată deoarece $M_0(p_1) = w(p_1, t_1)$; prin executarea lui t_1 se ajunge din M_0 la marcajul $M_1 = (1, 2, 0)$;
   - tranziția t_2 este validată deoarece $M_0(p_2) = w(p_2, t_2)$; prin executarea lui t_2 se ajunge din M_0 la marcajul $M_2 = (1, 0, 0)$;
   - tranziția t_3 este validată deoarece $M_0(p_1) = w(p_1, t_3)$ și $M_0(p_2) = w(p_2, t_3)$; prin executarea lui t_3 se ajunge din M_0 la marcajul $M_3 = (1, 0, 1)$.

3. Pentru rețeaua din fig. (c) cu marcajul inițial $M_0 = (2, 0, 0, 1)$, aplicarea regulii simple a tranziției conduce la:
   - tranziția t_1 este validată deoarece $M_0(p_1) = w(p_1, t_1)$; prin executarea lui t_1 se ajunge din M_0 la marcajul $M_1 = (0, 3, 0, 2)$;
   - tranziția t_5 este validată deoarece $M_0(p_4) = w(p_4, t_5)$; prin executarea lui t_5 se ajunge din M_0 la marcajul $M_2 = (4, 0, 0, 0)$.

2.3. Terminologii

O tranziţie fără nici o poziţie de intrare se numeşte ***tranziţie sursă*** .
O tranziţie fără nici o poziţie de ieşire se numeşte ***tranziţie receptor***.
Modul de operare al acestor tranziţii este următorul:

- *tranziţie sursă este necondiţionat validă* (fără a fi obligatoriu ca să se execute!). Executarea ei produce jetoane;
- executarea unei tranziţii receptor consumă *jetoane*, fără a produce.

Dacă o poziţie *p* este atât poziţie de intrare, cât şi poziţie de ieşire pentru o tranziţie *t* atunci *p* şi *t* formează o ***buclă automată***. O reţea Petri care *nu conţine* bucle autonome se numeşte *pură*.

O buclă autonomă poate fi întotdeauna transformată într-o buclă neautonomă prin adăugarea simultan a unei poziții și a unei tranziții formale.
Orice rețea impură poate fi transformată într-o rețea pură.

Exemplu de rețea impură care conține o buclă autonomă:

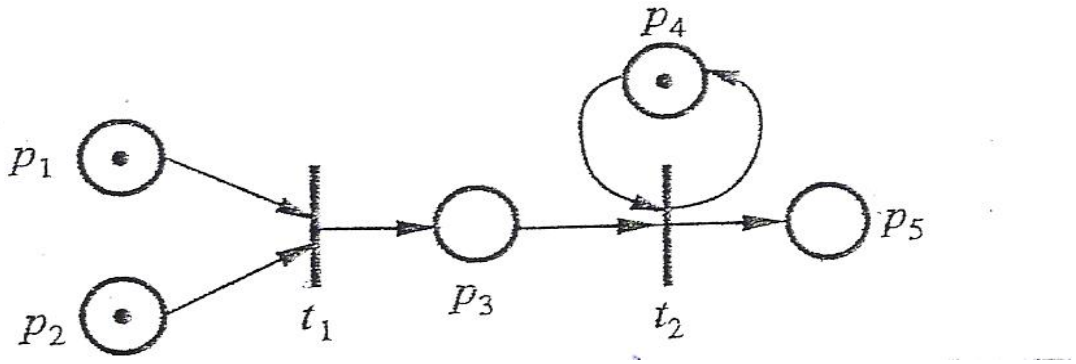


Fig. 2.6 Rețea impură.

Bucla autonomă poate fi transformată într-o buclă neautonomă prin adăugarea simultan a unei poziții și a unei tranziții formale (*dummy*), conform fig. 2.7, ceea ce transformă rețeaua impură într-o rețea pură.

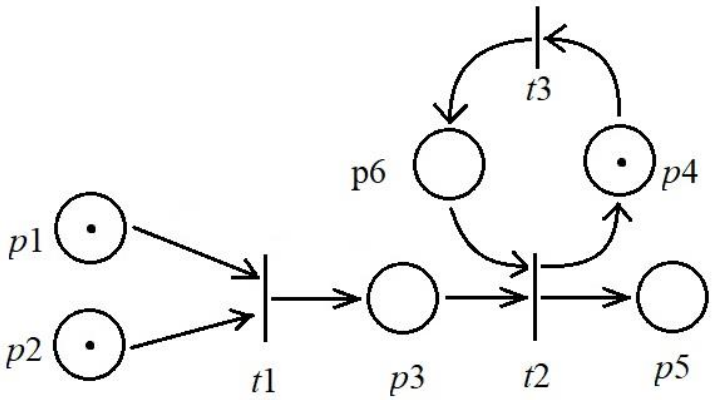


Fig. 2.7 Rețea pură.

Concret, în bucla autonomă (t_2, p_4) se adaugă tranziția t_3 și poziția p_6, bucla devenind neautonomă. O astfel de buclă neautonomă este referită simplu drept "buclă" sau "circuit", fără nici un alt atribut.

O rețea Petri se numește ***ordinară*** (*ordinary*) dacă toate arcele sale au pondere unitară.

Dacă într-o rețea Petri există cel puțin un arc a cărui pondere este mai mare decât 1, atunci se spune ca rețeaua respectivă este ***generalizată***.

Fie F mulțimea tuturor arcelor unei rețele Petri N. Se numesc *mulțime* ***predecesor*** (*pre-set*) și *mulțime succesor* (*post-set*) a ***tranziției*** t (fig. 2.8 (a)) două mulțimi de poziții definite prin:

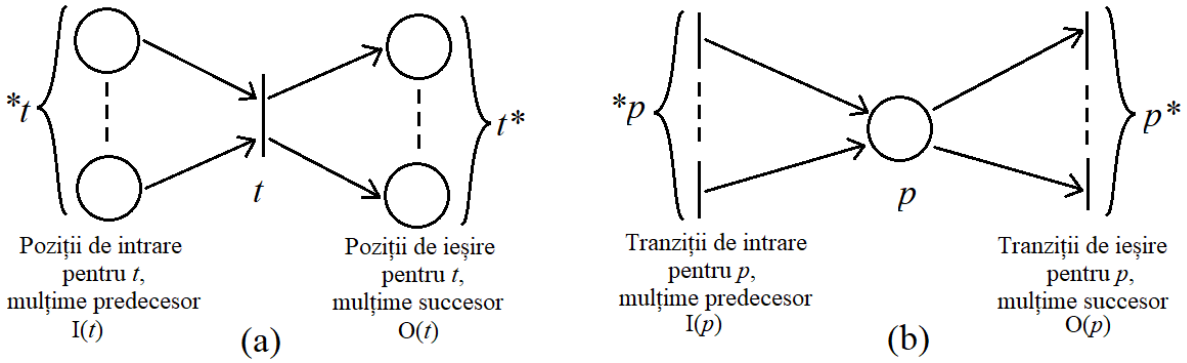
$^*t = \{p \mid (p, t) \in F\}$ = mulțimea tuturor pozițiilor de intrare ale lui $t \rightarrow I(t_j)$ – mulțimea pozițiilor de intrare în t_j și, respectiv, prin:

$t^* = \{p \mid (t, p) \in F\}$ = mulțimea tuturor pozițiilor de ieșire ale lui $t \rightarrow O(t_j)$ – mulțimea pozițiilor de ieșire din t_j.

Se numesc *mulțime predecesor* și ***mulțime succesor*** *a poziției* p (fig. (b)) două mulțimi de tranziții definite prin:

$^*p = \{t \mid (t, p) \in F\}$ = mulțimea tuturor tranzițiilor de intrare ale lui $p \rightarrow I(p_i)$ – mulțimea tranzițiilor de intrare în p_i și, respectiv, prin:

$p^* = \{t \mid (p, t) \in F\}$ = mulțimea tuturor tranzițiilor de ieșire ale lui $p \rightarrow O(p_i)$ – mulțimea tranzițiilor de ieșire din p_i.



(a) Mulțimile de poziții $^*t$ și t^* (b) Mulțimile de tranziții $^*p$ și p^*

Fig. 2.8 Ilustrarea grafică a mulțimilor predecesor și succesor.

Considerații:

În teoria rețelelor Petri netemporizate se consideră că executarea unei *tranziții nu consumă timp* și că *jetoanele pot rămâne în poziții pentru orice durată de timp* (oricât de mică sau oricât de mare). Întrucât executarea unei tranziții este instantanee, se consideră că tranzițiile se execută numai *secvențial*, adică nu se poate vorbi de două tranziții executate simultan (sau în parallel).

Aceste presupuneri fac ca modelul de tip *rețea Petri netemporizată* să fie utilizat numai pentru investigarea proprietăților *logice, calitative, care nu depind de timp.*

2.4. Dinamica rețelelor Petri

Modificând poziția jetoanelor într-o rețea Petri se realizează mecanismul tranzițiilor de stare în rețeaua respectivă. În urma validării unei tranziții, aceasta poate fi executată sau declanșată. Ca urmare a executării tranzițiilor validate a unei rețele Petri se poate stabili funcția tranzițiilor de stare a rețelei care reprezintă de fapt schimbările de stare.

Definiție – Dinamica rețelelor Petri

Funcția tranzițiilor de stare, $f : N^n \times T \rightarrow N^n$ a unei rețele Petri (P, T, F, W, X) marcate, este definită pentru tranziția $t_j \in T$ dacă și numai dacă $x(p_i) \geq w(p_i, t_j)$, $\forall\, p_i \in I(t_j)$. Dacă funcția tranzițiilor de stare $f(x, t_j)$ este definită, atunci se notează $x' = f(x, t_j)$, unde:

$$x'(p_i) = x(p_i) - w(p_i, t_j) + w(t_j, p_i),\ i = 1...n.$$

Exemplu:

Se ilustrează procesul de executare a tranzițiilor și schimbării stărilor rețelelor Petri din fig. 2.9.

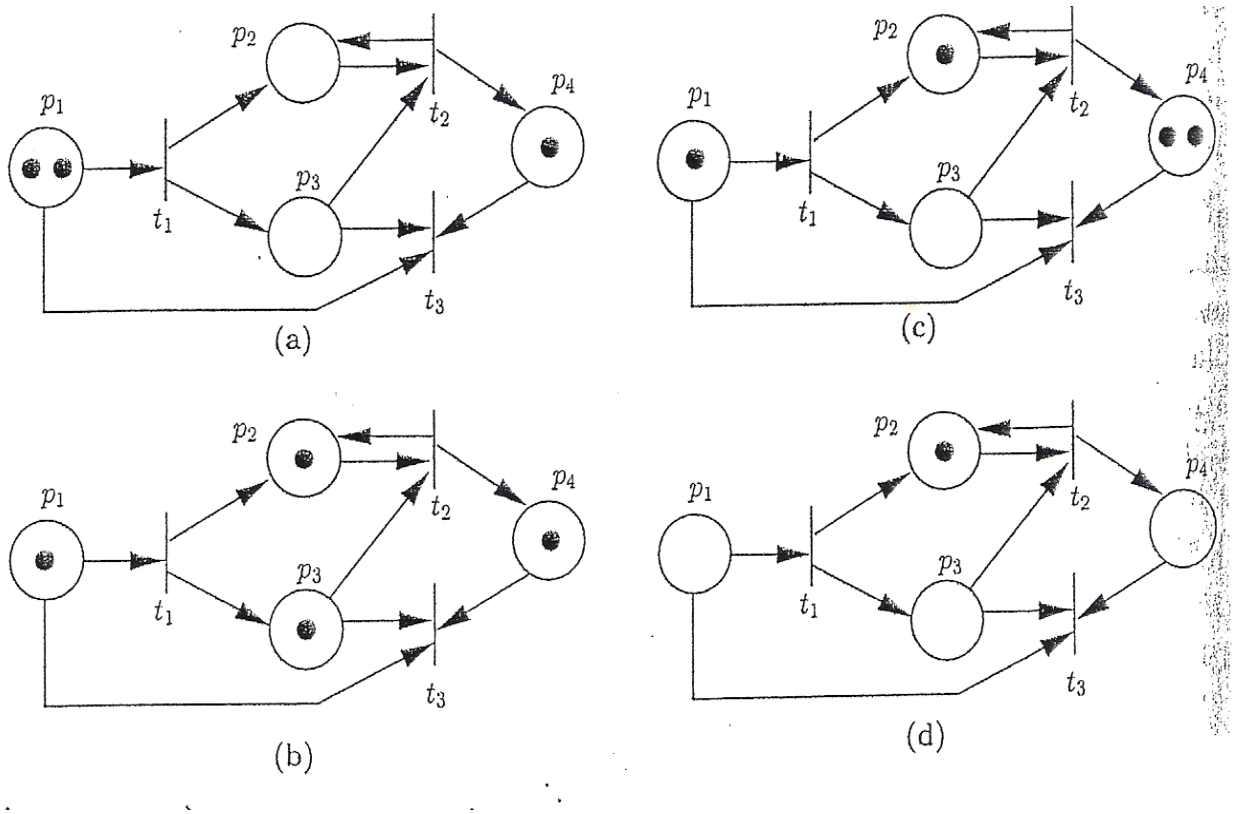


Fig. 2.9.

Starea inițială a rețelei este (a)

$M_0 / X_0 = [2\ 0\ 0\ 1]$.

Prin M_0/X_0 se specifică faptul că poate fi utilizată notația pentru starea inițială sau M_0 sau X_0. Se poate observa că doar t_1 este o tranziție validată, deoarece este necesar un singur jeton din poziția p_1, existând în fapt $x_0(p_1) = 2$.

Cu alte cuvinte: $x_0(p_1) > w(p_1, t_1)$ și condiția este satisfăcută pentru tranziția t_1.

Când t_1 este executată, un jeton este mutat din p_1, și un jeton este plasat în fiecare poziție p_2 și p_3 (fig. b).

Se poate aplica totodată direct și ecuația:

$$x'(p_i) = x(p_i) - w(p_i, t_j) + w(t_j, p_i);\ i=1, n,$$

$$x'(p_1) = x(p_1) - w(p_1, t_1) + w(t_1, p_1) = 2 - 1 + 0 = 1,$$
$$x'(p_2) = x(p_2) - w(p_2, t_1) + w(t_1, p_2) = 0 - 0 + 1 = 1,$$
$$x'(p_3) = x(p_3) - w(p_3, t_1) + w(t_1, p_3) = 0 - 0 + 1 = 1,$$
$$x'(p_4) = x(p_4) - w(p_4, t_1) + w(t_1, p_4) = 1 - 0 + 0 = 1.$$

deci noua stare este x_1 = [1, 1, 1, 1] (vezi fig. b).

- În continuare se presupune tranziția t_2 executată după obținerea stării x_1. Un jeton este deplasat din fiecare din pozițiile de intrare p_2 și p_3. Pozițiile de ieșire sunt p_2 și p_4. Prin urmare, un jeton este imediat plasat înapoi în p_2 și un jeton este adăugat în p_4.

Noua stare este:

x_2 = [1, 1, 0, 2] (vezi fig. c).

În această stare, t_2 și t_3 nu mai sunt validate, dar t_1 este.
Prin aplicarea ecuației dinamicii rețelelor Petri se obține:

$$x'(p_1) = x(p_1) - w(p_1, t_2) + w(t_2, p_1) = 1 - 0 + 0 = 1,$$
$$x'(p_2) = x(p_2) - w(p_2, t_2) + w(t_2, p_2) = 1 - 1 + 1 = 1\ ,$$
$$x'(p_3) = x(p_3) - w(p_3, t_2) + w(t_2, p_3) = 1 - 1 + 0 = 0,$$
$$x'(p_4) = x(p_4) - w(p_4, t_2) + w(t_2, p_4) = 1 - 0 + 1 = 2.$$

- Fie din nou starea x_1 din fig. b, unde toate trei tranzițiile sunt validate. În loc de executarea lui t_2, se execută t_3. Se va muta câte un jeton din fiecare poziție de intrare p_1, p_3 și p_4.

Se observă că nu există poziții de ieșire.

Noua stare x_2' este:

$x_3 = x_2'$ = [0, 1, 0, 0] (ca în fig. d)

Se observă că nici o tranziție nu este în acest moment validată. Nu mai e posibilă nici o schimbare de stare și [0, 1, 0, 0] este o stare blocată a rețelei.
Numărul de jetoane nu este conservat în această rețea Petri deoarece x_0 conține 3 jetoane, în timp ce x_1 și x_2 conțin fiecare 4 jetoane, iar x_2' conține 1 jeton.

2.5. Reţele Petri cu capacitate finită şi infinită

Pentru regula de validare a unei tranziţii s-a presupus că fiecare poziţie poate conţine un număr *nelimitat* de jetoane. O astfel de rețea Petri se numește cu *capacitate infinită*. În modelarea sistemelor fizice este firesc a considera o limită superioară a numărului de jetoane pe care îl poate conţine fiecare poziţie. O astfel de rețea se numeşe cu *capacitate finită*.

Într-o rețea cu capacitate finită (N, M_0), fiecărei poziţii p i se asociază *capacitatea poziţiei*, notată $K(p)$, definită ca numărul maxim de jetoane ce pot fi conţinute în p.

Într-o reţea cu capacitate finită, pentru validarea unei tranziţii t este necesară următoarea condiţie suplimentară: numărul de jetoane în fiecare poziţie de ieşire p a lui t nu poate să depăşească capacitatea poziţiei respective, K(p), atunci când t se execută.

Astfel, o tranziție t_j a unei rețele Petri cu capacitate finită se numește validată dacă:

$x(p_i) \geq w(p_i, t_j)$ pentru toate $p_i \in I(t_j)$

și $K(p_i) \geq w(t_j, p_i) + x(p_i)$ pentru toate $p_i \in O(t_j)$.

În acest caz regula tranziţiei se va numi **regula strictă a tranziţiei** spre a o deosebi de cea enunţată în paragraful 2.2, care este referită drept regula simplă a tranziţiei.

Fiind dată o rețea Petri de capacitate finită (N, M_0) este posibil de aplicat fie regula strictă a tranziţiei direct pentru reţeaua (N, M_0), fie regula simplă a tranziţiei pentru o *reţea transformată adecvat*, notat (N', M'_0).

Presupunând că reţeaua N este pură, următorul *algoritm* permite construcţia reţelei (N', M'_0) plecând de la (N, M'_0), prin **metoda poziţiilor complementare**:

Pas 1. Pentru fiecare poziţie p, se adauă o poziţie complementară p', al cărei marcaj iniţial este dat de:

$$M'_0(p') = K(p) - M_0(p).$$

Pas 2. Între fiecare tranziţie t şi unele poziţii complementare p' se trasează arce suplimentare, (t, p') sau (p', t), cu ponderile w(t, p') = w(p, t), respectiv w(p', t) = w(t, p), astfel încât suma jetoanelor în poziţia p şi în poziţia complemantară p' să fie egală cu capacitatea K(p), atât înainte, cât şi după executarea tranziţiei t (adică să se asigure satisfacerea condiţiei M(p)+M'(p')=K(p)).

Se subliniază faptul că *metoda nu adaugă tranziţii suplimentare*.

Exemplu:

Se consideră rețeaua Petri (N, M_0) din fig. 2.10 (a).

Se pune problema transformării rețelei (N, M_0), într-o rețea (N', M'_0) utilizând metoda pozițiilor complementare.