CONTROLUL PROCESELOR – SARCINILOR CONCURENTE

*Obiective

- Acest capitol este dedicat tratării separate a celor mai reprezentative aspecte legate de controlul şi gestiunea sarcinilor concurente.
- Se pune un accent deosebit pe prezentarea
  - unor algoritmi de excludere mutuala
    - context centralizat
    - context distribuit
  - analiza situaţiilor de blocare
  - Sincronizarea sarcinilor.
- Problemele tratate în acest capitol asigură baza teoretică pentru analiza sau dezvoltarea unor sisteme software care sa lucreze
  - în regim de multiprogramare
  - multiprelucrare
- atât pentru sisteme monoprocesor, cât şi pentru sisteme cu prelucrare paralelă

Definiţii şi concepte de bază

- Activităţile paralele se pot desfăşura atât în sisteme de tip MIMD, SIMD cât şi în cadrul sistemelor convenţionale, de tip SISD.
- Un exemplu de activităţi paralele într-un sistem de calcul conventional de tip SISD, Von Neumann îl constituie efectuarea simultană a operaţiilor din UCP, cu cele din subsistemul de intrări/ieşiri prin DMA sau canal de I/E.
- De asemenea, activităţi paralele se întâlnesc la nivelul microoperaţiilor care constituie o microinstrucţiune în cadrul unei unităţi de comandă microprogramată.

În general pentru creşterea performanţelor unui sistem de calcul se prevăd mai multe procesoare de diferite tipuri, cum ar fi:

- procesoare centrale de prelucrare;
- procesoare de intrări/ieşiri;
- procesoare specializate (coprocesoare matematice, coprocesoare neurale, etc).

- Pe de altă parte mai multe programe sau părţi ale unui aceluiaşi program pot fi executate în paralel.
- Comunicaţia între acestea poate fi asigurată printr-un schimb de mesaje.
- Chiar dacă există un singur procesor (nu o structură MIMD), care este partajat între mai multe programe, vom admite că, din punct de vedere logic aceste programe se execută în paralel disputându-şi accesul la diferite resurse fizice ale sistemului.

Sarcina -proces secvenţial

Sarcina - proces secvenţial este o activitate care constă din execuţia, pe un procesor, a unui grup de instructiuni (de obicei indivizibil) asociat unui set de date specificat.

- Deşi pare că fiecare sarcina are procesorul şi datele sale, în realitate mai multe sarcini pot utiliza în comun
  - un procesor
  - o secvenţă de cod
  - o structură de date.

O *sarcină* poate fi specificată prin relaţia sa cu exteriorul:

- intrările necesare;
- funcţia executată;
- ieşirile generate;
- starea la un moment dat;
- timpul de execuţie.

- Comportarea intrinsecă, internă, nu va fi specificată decât în situaţii excepţionale, pentru a facilita înţelegerea interacţiunii cu celelalte sarcini.
- Une sarcini i se asociază două evenimente:
  - SI - iniţiere sarcină;
  - SF - terminare sarcină.
- Dacă notăm cu t(·) timpul de apariţie a unui eveniment, vom considera că:

  t(SF) - t(SI) $\neq$ 0 şi finit

cu condiţia ca toate resursele necesare execuţiei lui S să fie disponibile în momentul t(SI)

Vom considera sarcinile neinterpretate, ceea ce este echivalent cu faptul că pentru o mulţime dată de sarcini să ne intereseze secvenţele evenimentelor de iniţiere şi terminare fără să ne preocupe particularităţile de execuţie ale sarcinilor.

În ceea ce priveşte granularitatea unei sarcini, aceasta poate fi:

- program -JOB;
- modul de program - task;
- procedură - sarcina;
- secventa - thread
- instrucţiune;
- microinstrucţiune;
- microoperaţie.

Sistemul de calcul în care se execută sarcinile constă dintr-o mulţime de resurse:

- procesoare;
- dispozitive de I/E;
- biblioteci de programe;
- proceduri;
- variabile;
- fişiere de date, etc.

şi este caracterizat de o mulţime Σ de stări.

În funcţie de granularitatea sarcinilor, procesorul poate fi:

- calculator universal;
- UCP într-un sistem multiprocesor;
- UAL;
- unitate de comandă;
- microsecvenţiator într-o unitate de comandă.

Atât posibilităţile sistemului, cât şi funcţiile sarcinilor date sunt reprezentate prin tranziţii permise în Σ : $s_0 s_1 s_2 \ldots\ldots\ldots s_n$–spatiul starilor,

de forma $s_i \rightarrow s_j$, $(s_i, s_j \in \Sigma)$ definite pentru iniţierea şi terminarea sarcinilor.

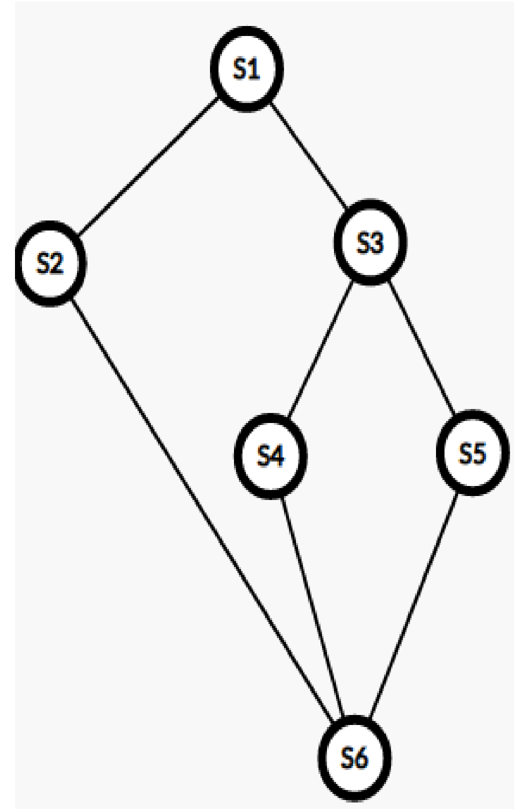
Iniţierea unui sarcina corespunde unei tranziţii de stări care reflectă:

- preluarea resurselor necesare;
- iniţializarea stării sistemului (a resurselor);
- citirea parametrilor de intrare.

Terminarea unei sarcini corespunde unei tranziţii de stări care reflectă:

- eliberarea sau suspendarea temporară a utilizării resurselor;
- scrierea parametrilor de ieşire;
- dacă valori ale parametrilor sau stări interne sunt asociate cu resursele, terminarea unei sarcina trebuie însoţită de valoarea stării resurselor.

Fie ***S*** = $\{ S_1, S_2, \ldots, S_n \}$ o mulţime de sarcini iar **<** o relaţie de ordine parţială, nereflexivă, pe ***S*** (denumită şi relaţie de precedenţă pe mulţimea de sarcini ***S***).
Perechea **C** = (***S*** , **<**) este denumită **sistem de sarcini.**



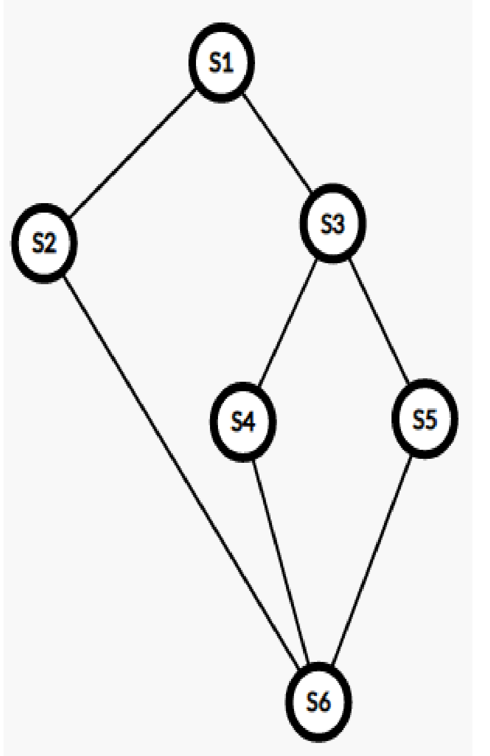
- Relaţia **<** reprezintă precedenţele în execuţia sarcinilor.

Astfel, S_i **<** S_j

- înseamnă că S_i trebuie terminată înainte de iniţierea lui S_j.

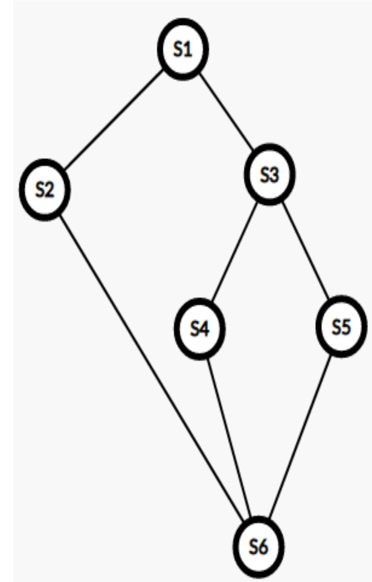
Sarcinile din ***S*** sunt independente dacă pentru **C** = (***S*** , **<**)
avem **<** = Φ; unde Φ este mulţimea vidă.

- O reprezentare grafică convenabilă a unui sistem de sarcini se obţine utilizând grafuri aciclice direcţionate, *GAD*.
- Sracinile sunt reprezentate prin noduri iar relaţia de precedenţă prin mulţimea arcelor.
- Relaţia de precedenţă dată de mulţimea arcelor poate fi definită ca fiind cea mai mică relaţie în ***S*** a cărui închidere prin tranzitivitate este **<**.
- Deoarece **<** reprezintă o ordonare în timp a sarcinilor, vom spune că două sarcini S_i şi S_j, pot fi concurente dacă şi numai dacă ele sunt independente, adică între ele nu există o relaţie de precedenţă.



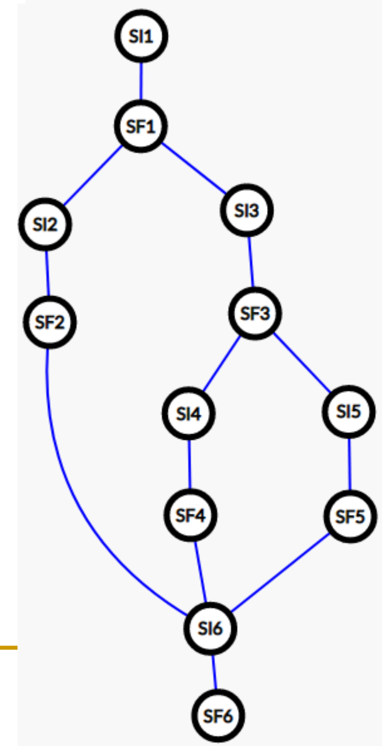
*Exemplu:

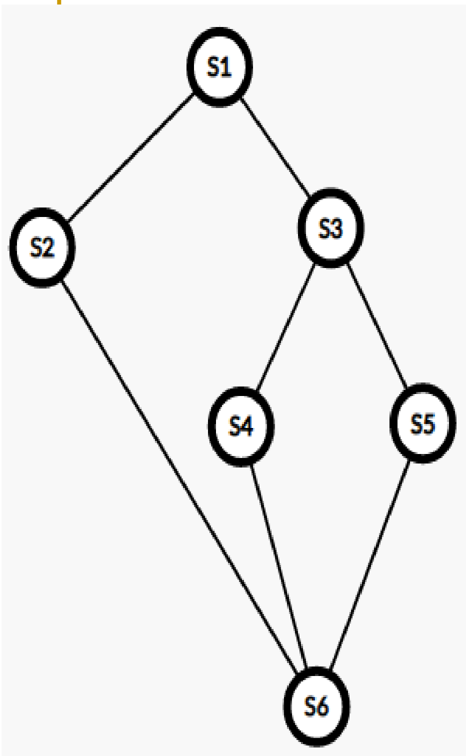
- Două sarcini S_i si S_j sunt independente dacă S_i nu este nici predecesor, nici succesor al lui S_j.
- O sarcină S din sistemul de sarcini **C** este pe nivelul k, sau are nivelul k, dacă cea mai lungă cale de la S la un nod terminal are lungimea k.
- Un nod terminal are nivelul 1.

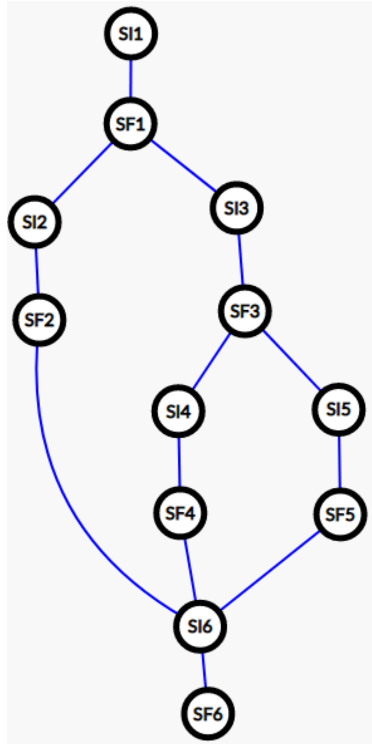


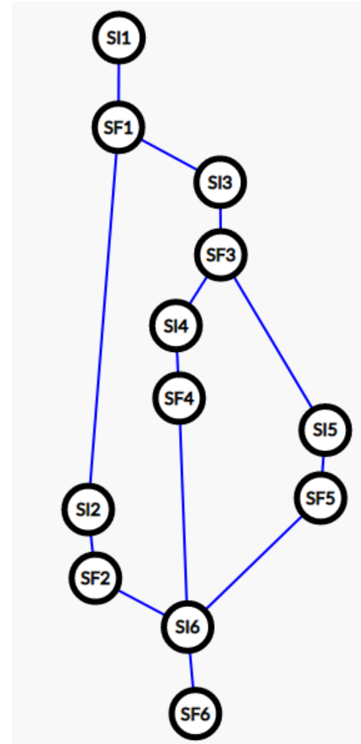
O secvenţă de execuţie a unui sistem de n sarcini, **C** = (***S***, **<**) este orice şir $\alpha = a_1 a_2 ......a_{2n}$ de evenimente de iniţiere şi terminare a sarcinilor cu respectarea relaţiilor de precedenţă impuse de **<**.

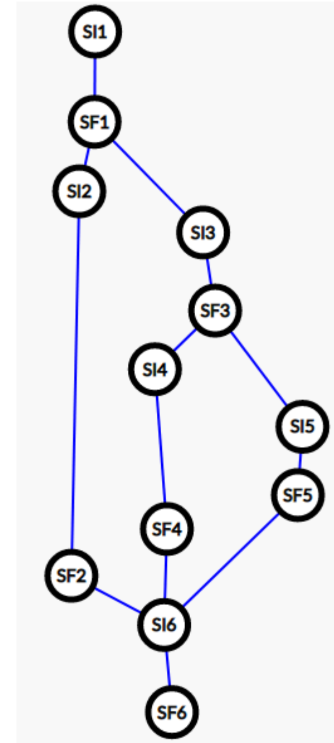
- Altfel spus:
  - pentru orice S Є ***S***, SI – S initiat şi SF - S terminat apar o singură dată în α ;
  - dacă a_i= SI şi a_j= SF, atunci i < j ;
  - dacă a_i= SF_k şi a_j= SI_l ,cu S_k **<** S_l atunci i < j ;









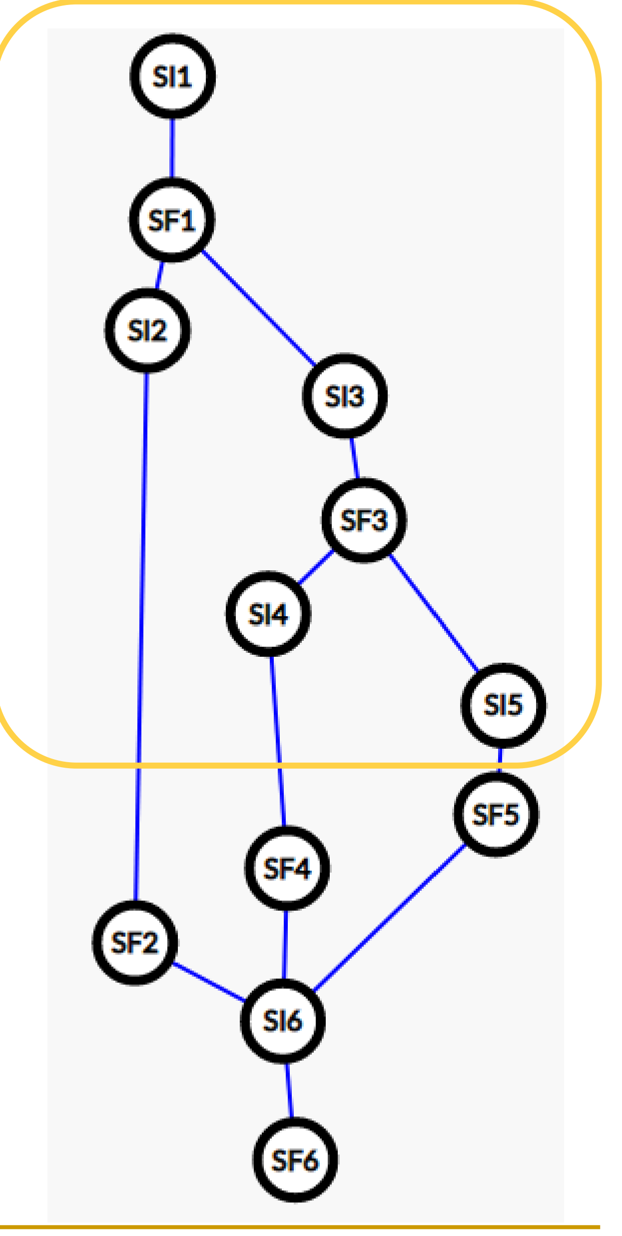


Sistem de sarcini reprezentat prin graf

Exemple de secvenţe de execuţie pentru sistemul de sarcini

- C=(S, $<$)
- S={S1,S2, S3,S4,S5,S6}
- reprezentat prin graful
- $\alpha_1 = SI_1\ SF_1\ SI_3\ SF_3\ SI_4\ SF_4\ SI_5\ SF_5\ SI_2\ SF_2\ SI_6\ SF_6$
- $\alpha_2 = SI_1\ SF_1\ SI_2\ SI_3\ SF_3\ SI_4\ SI_5\ SF_5\ SF_4\ SF_2\ SI_6\ SF_6$
- $\alpha_3 =$

secvenţă de execuţie parţială



O secvenţă de execuţie parţială este orice prefix al unei secvenţe de execuţie.

Mulţimea secvenţelor de execuţie reprezintă mulţimea tuturor secvenţelor de evenimente (de iniţiere şi terminare) care conduc la finalizarea lui **C** respectându-se relaţia **<**.

Un sarcină S este activă după o secvenţă de execuţie parţială

$\alpha_p = a_1 a_2 \ldots\ldots a_k$, dacă există $i \le k$ astfel că

a_i= SI, dar pentru orice $j \le k$, $a_j \neq$SF

Ex

$\alpha_p = SI_1\ SF_1\ SI_2\ SI_3\ SF_3\ SI_4\ SI_5$

- $\alpha_2 = SI_1\ SF_1\ SI_2\ SI_3\ SF_3\ SI_4\ SI_5\ SF_5\ SF_4\ SF_2\ SI_6\ SF_6$

Fie Σ spaţiul stărilor pentru un sistem de sarcini.

- Secvenţa stărilor corespunzătoare secvenţei de execuţie

$$\alpha = a_1\ a_2 \ldots\ldots a_{2n}$$

este dată prin:

$$\sigma = s_0 s_1 s_2 \ldots\ldots\ldots s_{2n}, \quad s_j \in \Sigma,\ \ 0 \leq j \leq 2n\ ,$$

iar s_0 este starea iniţială dată.

- Tranziţia stării definită de evenimentul a_i este reprezentată de:

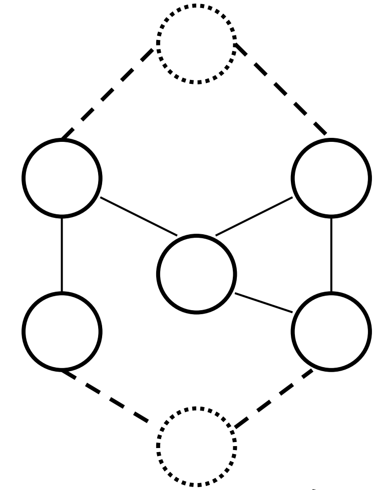
$$s_{i-1} \longrightarrow s_i$$

- O tranziţie pentru o pereche de stări-eveniment (s,a) va fi definită numai dacă evenimentul a poate să apară când sistemul este

în starea s.

Un sistem de sarcini reprezentat printr-un graf cu un sigur nod iniţial şi un singur nod terminal se numeşte închis.

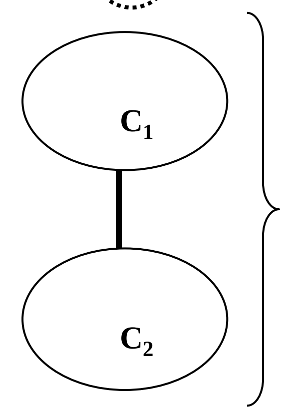
- Dacă sistemul de n sarcini nu este închis se pot prevedea sarcinile S_0 şi S_{n+1} inoperante pentru a obţine un sistem închis.



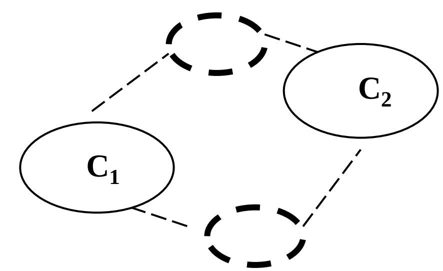
- Două sisteme de sarcini închise $\mathbf{C}_1$ şi $\mathbf{C}_2$ pot fi concatenate ($\mathbf{C}_1.\mathbf{C}_2$), graful asociat obţinându-se prin introducerea unui arc de la nodul terminal al lui $\mathbf{C}_1$ la nodul iniţial al lui $\mathbf{C}_2$.
- O secvenţă de execuţie a lui ($\mathbf{C}_1.\mathbf{C}_2$) este orice şir

$$\alpha = \alpha_1.\alpha_2$$

de secvenţe de execuţie α_1 a lui $\mathbf{C}_1$, α_2 a lui $\mathbf{C}_2$.



Combinarea paralelă a două sisteme de sarcini $\mathbf{C}_1$ şi $\mathbf{C}_2$, $\mathbf{C}_1 \mid \mathbf{C}_2$, care nu au sarcini în comun,($\forall S_i \in \mathbf{C}_1$ şi $\forall S_j \in \mathbf{C}_2$, $S_i \neq S_j$) constă în simpla lor reuniune.



- Graful lui $\mathbf{C}_1 \mid \mathbf{C}_2$ are ca subgrafuri disjuncte grafurile lui $\mathbf{C}_1$ şi $\mathbf{C}_2$.

Orice sarcină din $\mathbf{C}_1$ este independentă de orice sarcina din $\mathbf{C}_2$.
Dacă $a = a_1\ a_2......a_{2m}$ şi $b = b_1\ b_2......b_{2n}$ sunt secvenţe de execuţie pentru $\mathbf{C}_1$ respectiv pentru $\mathbf{C}_2$, o secvenţă de execuţie a lui $\mathbf{C}_1 \mid \mathbf{C}_2$ se formează astfel:

$$c = c_1 c_2.........c_{2(m+n)}$$

unde :

$c_1 = a_1 \sim b_1$ (fie a_1 fie b_1)

dacă $a_1\ a_2......a_i$ şi $b_1\ b_2......b_j$ sunt elemente ale lui $c_1\ c_2......c_{i+j}$ cu $i+j < 2(m+n)$

atunci :

$$c_{i+j+1} = a_{i+1} \sim b_{j+1} \quad \text{(fie } a_{i+1} \text{ fie } b_{j+1} \text{)}$$

Combinarea paralelă a secvenţelor de execuţie apare frecvent în proiectarea sistemelor, în special a sistemelor cu prelucrare paralelă, uneori sub o formă diferită de cea prezentată anterior în sensul că unele sisteme implică secvenţe repetitive.

- Sistemele care conţin secvenţe repetitive pot fi modelate fie prin grafuri aciclice fie prin grafuri ciclice.
- Astfel, se pot modela k cicli ai unui sistem **C** prin concatenarea

$$\mathbf{C}^k = \mathbf{C}_1 \cdot \mathbf{C}_2 \cdot \ldots\ldots \cdot \mathbf{C}_k$$

- O secvenţă de execuţie α a lui $\mathbf{C}^k$ este de forma:

$$\alpha = \alpha_1 \alpha_2 \ldots\ldots \alpha_k$$

unde α_i este o secvenţă de execuţie a lui **C** pentru iteraţia i.

> Un sistem de sarcini se numeşte ciclic dacă este de forma $\mathbf{C}^k$ pentru k > 1 , sau dacă este o combinare paralelă de sisteme închise în care cel puţin unul este ciclic.

Proprietatea de determinare a sistemelor de sarcini

Definiţie:

Sistem determinat (sistem de sarcini functional) este un sistem format din sarcini care se execută în paralel şi cooperează pentru realizarea unor operaţii logice şi de calcul, care produce acelaşi rezultat indiferent de durata de execuţie a fiecărei sarcini **independente**, sau de ordinea în care acestea se executa.

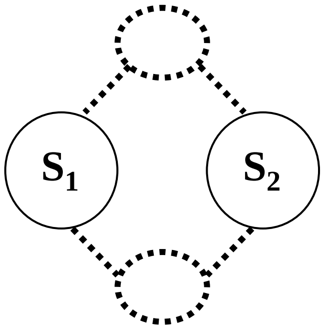
Definiţie:

Un sistem de sarcini este nedeterminat dacă rezultatele produse de sarcini independente depind de ordinea în care acestea se executa.

- Exemplu de sistem nedeterminat:

    S_1: $R_1 \leftarrow$ BUSFN (M; DCD (ADR))

    S_2: M * DCD (ADR) $\leftarrow R_2$



S_1 - citeşte din locaţia de memorie de la adresa ADR;

S_2 - scrie în locaţia de memorie de la adresa ADR.

$\alpha_1 = SI_1\ SF_1\ SI_2\ SF_2$

se va citi valoarea R_1 si in celula de memorie va ramane valoarea R_2

$\alpha_2 = SI_2\ SF_2\ SI_1\ SF_1$

in celula de memorie va ramane valoarea R_2 si se va **citi valoarea R2**

Produc rezultate diferite

- Nedeterminarea se poate rezolva introducând o relaţie de precedenţă adecvată între sarcinile care erau independente.

- Pentru a stabili condiţiile necesare şi suficiente ca un sistem să fie determinat, vom considera un model simplificat în care sistemul fizic este privit ca o mulţime ordonată de locaţii de memorie:

$$\mathbf{M} = (M_1, M_2, ..., M_m)$$

ce conţin orice valoare dintr-o mulţime de valori **V**.

- Stările sistemului vor fi definite de valorile care se găsesc în memorie la un moment dat:

De exemplu dacă:

$$\alpha = a_1 a_2 .... a_{2n} \text{ şi}$$

$$\sigma = \sigma_0 \sigma_1 \sigma_2 .... \sigma_{2n}$$

reprezintă o secvenţă de execuţie, respectiv secvenţa de stări corespunzătoare,

iar $M_i(k)$reprezintă valoarea din celula M_i imediat după evenimentul a_k

starea sistemului va fi:

$$\sigma_k = [M_1(k), M_2(k), ..., M_m(k)]$$

Mulţimea tuturor stărilor poate fi definită astfel:

$\boldsymbol{\Sigma} = \mathbf{V}^m$ $\mathbf{V}^m = [\,\mathbf{V} \times \mathbf{V} \times ....\times \mathbf{V}\,]$, produs cartezian.

- Pentru a formaliza efectul execuţiei unei sarcini asupra celulelor de memorie vom considera că fiecărei sarcini S i se asociază funcţia:

$$f_S : V^d \longrightarrow V^r$$

unde:

$d = |D_S|$ cardinalul domeniului valorilor de intrare, D_S;

$r = |R_S|$ cardinalul domeniului valorilor de ieşire, R_S.

- Pentru o stare iniţială $\mathfrak{s}_0$ şi o secvenţă de execuţie α, secvenţa corespunzătoare de stări: $\sigma = \mathfrak{s}_0\mathfrak{s}_1........\mathfrak{s}_{2n}$ este definită în felul următor :

Fie $D_S = (M_{x1}, M_{x2}, ..., M_{xd})$ domeniul valorilor de intrare;

$R_S = (M_{y1}, M_{y2}, ..., M_{yr})$ domeniul valorilor de ieşire;

dacă a_{k+1} = SI, atunci $M_i(k+1) = M_i(k)$, $1 \le i \le m$;

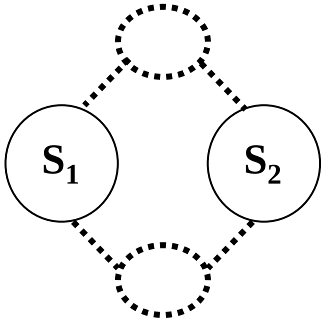
dacă a_{k+1} = SF, şi a_l = SI , $l \le k$, atunci stările locaţiilor de memorie din domeniul de valori al lui S la momentul k+1 sunt:

$[M_{y1}(k+1), M_{y2}(k+1), ..., M_{yr}(k+1)] = f_S(M_{x1}(l), M_{x2}(l), ..., M_{xd}(l))$ şi

$M_i(k+1) = M_i(k)$ pentru $(\forall)\ M_i \notin R_S$

- Altfel spus, dacă a_{k+1} este un eveniment de iniţiere nu apare o schimbare a stării, adică $s_{k+1} = s_k$.
- Dacă însă a_{k+1} este un eveniment de terminare a lui S, $s_{k+1} \neq s_k$ doar în domeniul R**s**, noile valori din R**s** fiind determinate de f_S pe baza valorilor din domeniul de definiţie din momentul imediat precedent iniţierii lui S .
- Secvenţa de stări $\sigma = s_0 s_1 ...... s_{2n}$ ce rezultă dintr-o secvenţă de execuţie, poate fi reprezentată sub forma unui tablou de dimensiuni m * (2n+1) având
- pe linii celulele de memorie M, iar
- pe coloane stările.

Sistem de sarcini nedeterminat



*Exemplu:

Să considerăm un sistem format din două sarcini, iar relaţia de precedenţă fiind mulţimea vidă

$$\mathbf{C} = (\{S_1,S_2\}, \Phi)$$
$$\mathbf{M} = (M_1 , M_2)$$
$$D_{S1} = D_{S2} = R_{S1} = R_{S2} = \mathbf{M}$$

Sistemului i se atribuie două interpretări:

Sarcini	Interpretare 1	Interpretare 2
S_1	$(M_1,M_2) \leftarrow (1,M_2)$	$(M_1,M_2) \leftarrow (M_1+M_2,M_2)$
S_2	$(M_1,M_2) \leftarrow (2,M_2)$	$(M_1,M_2) \leftarrow (M_1,M_1+M_2)$

Valorile iniţiale corespunzătoare stării iniţiale s_0: $M \leftarrow (1,2)$

Fie două secvenţe de execuţie α_1 şi α_2

$$\alpha_1 = SI_1\, SF_1\, SI_2\, SF_2 \quad \text{şi}$$
$$\alpha_2 = SI_2\, SF_2\, SI_1\, SF_1$$

Comportarea celor două secvenţe de execuţie

Să analizăm comportarea celor două secvenţe de execuţie conform cu cele două interpretări ale sistemului:

	α_1 = $SI_1\ SF_1\ SI_2\ SF_2$	α_2 = $SI_2\ SF_2\ SI_1\ SF_1$
	$\sigma = s_0\ s_1\ s_2\ s_3\ s_4$	$\sigma = s_0\ s_1\ s_2\ s_3\ s_4$
Interpretare 1	M_1 1 1 1 1 2	M_1 1 1 2 2 1
	M_2 2 2 2 2 2	M_2 2 2 2 2 2
Interpretare 2	M_1 1 1 3 3 3	M_1 1 1 1 1 4
	M_2 2 2 2 2 5	M_2 2 2 3 3 3

Se observă că sistemul de sarcini C nu este determinat pentru nici una din cele două interpretări .

Reprezentare prin secventa de valori

- O reprezentare mai convenabilă constă din secvenţa de valori pe care un sistem de sarcini dat C o înscrie într-o celulă de memorie Mi, în timpul unei secvenţe de execuţie α .
- Această reprezentare se notează sub formă vectorială:

$$V(M_i,\alpha) = (V_1,V_2,......,V_p)$$

Considerând

$\alpha = a_1a_2........a_{2n}$ secvenţa de execuţie

$\sigma = s_0s_1....... s_{2n},$ secvenţa de stări corespunzătoare,

ε= Φ şirul vid, secvenţa de valori corespunzătoare se defineşte astfel:

$V(M_i,\varepsilon) = M_i(\Phi)$ în lipsa unei secvenţe de execuţie, iar

$V(M_i, a_1a_2 \ldots a_k)=$ <u>dacă</u> a_k= SF şi $M_i \in R_S$ (domeniul unde S produce valori)

<u>Atunci</u> = $(V(M_i, a_1a_2 \ldots a_{k-1}), M_i(k))$

<u>Altfel</u> nu se modifică deci este $V(M_i, a_1 \ldots a_{k-1})$

nu se mai adaugă o nouă valoare

- Se iau în considerare numai valorile înscrise în celulele de memorie din domeniul de valori şi numai la terminarea S.

Obs. V(M_i, α)= ..... V(M_j, α)= .... pot avea dimensiuni diferite

- Trebuie notat că V(M_i,α) şi V(M_j,α) nu trebuie să fie de aceeaşi lungime deoarece M_i poate fi în R pentru anumite sarcini
- iar M_j poate fi în R pentru altele (în număr diferit).

Pentru secvenţa de valori: V(M_i,α) = (v_1,v_2,......,v_p) se defineşte *valoarea finală* F(M_i,α) = v_p.

Rezultă că pentru secvenţa de execuţie partiala

$\alpha = a_1 a_2 ... a_k$

σ_k = [F(M_1,α),F(M_2,α),...,F(M_m,α)]

	α	a1	a2	a3	a4	a5	....	ak	....	a2n
	σ0	σ1	σ2	σ3	σ4	σ5	....	σk	....	σ2n
M1										
M2										
M3										
....										
Mi										
....										
Mj										
....										
Mm										

Exemplu: Să reconsiderăm exemplul precedent:

* Secvenţa de stări poate fi reprezentată astfel printr-un tablou:

	α_1 = SI_1 SF_1 SI_2 SF_2			α_2 = SI_2 SF_2 SI_1 SF_1		
	V(M,ε)	V(M,a_1a_2)	V(M,$a_1..a_4$)	V(M,ε)	V(M,a_1a_2)	V(M,$a_1..a_4$)
Interpretare 1	M_1 1	1	2	M_1 1	2	1
	M_2 2	2	2	M_2 2	2	2
Interpretare 2	M_1 1	3	3	M_1 1	1	4
	M_2 2	2	5	M_2 2	3	3

Interpretarea 1 este nedeterminată deoarece S_1 şi S_2 sunt în dispută pentru a scrie în M_1, iar
în interpretarea 2 o sarcina scrie într-o celulă citită de alta sarcina.

	α_1 = SI_1 SF_1 SI_2 SF_2	α_2 = SI_2 SF_2 SI_1 SF_1
	σ = s_0 s_1 s_2 s_3 s_4	σ = s_0 s_1 s_2 s_3 s_4
Interpretare 1	M_1 1 1 1 1 2	M_1 1 1 2 2 1
	M_2 2 2 2 2 2	M_2 2 2 2 2 2
Interpretare 2	M_1 1 1 3 3 3	M_1 1 1 1 1 4
	M_2 2 2 2 2 5	M_2 2 2 3 3 3

- Un sistem de sarcini **C** = (***S***, **<**) este neinterpretat dacă se cunosc doar relatia de precedenta **<** , domeniile de definitie D şi de valori R pentru fiecare sarcina.
- O interpretare pentru **C** constă din specificarea funcţiei f_S pentru fiecare S_i Є ***S*** .

În continuare vom considera sisteme de sarcini **neinterpretate.**

Definiţie:
Un sistem de sarcini ***C*** *este determinat* dacă pentru orice stare iniţială s_0 dată, **V(M$_i$,α) = V(M$_i$,α'),** $1 \le i \le m$, pentru toate secvenţele de execuţie α ,….α' din **C**.

(Secvenţele de valori depind numai de valorile iniţiale în s_0.)

Definiţie:
Sarcinile S şi S' sunt neinterferente dacă:

S este succesor sau predecesor a lui S', sau

$R_S \cap R_{S'} = R_S \cap D_{S'} = D_S \cap R_{S'} = \Phi$

- Mulţimea ***S*** = { $S_1, S_2, \ldots S_n$ } se spune că este formată din sarcini mutual neinterferente dacă pentru $\forall$ i , j (i $\neq$ j) S_i si S_j sunt neinterferente.
- Pentru a arăta că un sistem de sarcini mutual neinterferente este determinat vom prezenta *teorema de suficienţă şi necesitate.*
- Înainte de a prezenta această teoremă vom stabili o lemă utilizată în cadrul teoremei de necesitate şi suficienţă.

Lemă:
Fie **C** = (**<** , α) un sistem de n sarcini mutual neinterferente iar S o sarcina terminala a lui **C**. $(R_S \cap R_{S'} = R_S \cap D_{S'} = D_S \cap R_{S'} = \Phi)$
Dacă $\alpha = \beta_1$ SI β_2 SF β_3 este o secvenţă de execuţie validă a lui **C**, atunci
$\alpha' = \beta_1 \ \beta_2 \beta_3$ SI SF este de asemenea o secvenţă de execuţie a lui **C** pentru $\mathcal{S}_0$ dată,

$$V(M_i,\alpha) = V(M_i,\alpha'), \ \forall \ 1 \le i \le m$$

Justificare :

- S nu are succesori în **C** conform cu (**<** , α') satisface relaţiile de precedenţă din **C** şi deci trebuie să fie o secvenţă de execuţie validă.
- S scrie numai în celulele ce aparţin lui R_S şi deoarece pentru orice S' iniţiată în β_3, după terminarea lui S în α, $R_S \cap D_{S'} = \Phi$ fiind neinterferente, orice astfel de sarcini S' găsesc aceleaşi valori în $D_{S'}$ atât pentru α cât şi pentru α'.
- Deci pentru $M_i \notin R_S$ rezultă $V(M_i,\alpha)=V(M_i,\alpha')$

- Pentru orice $M_j \in D_S$, $V(M_j,\beta_1)=V(M_j\ ,\ \beta_1\beta_2\beta_3)$ deoarece nici o sarcina S' din $\beta_1\beta_2\beta_3$ nu scrie în D_S deoarece $R_{S'} \cap D_S = \Phi$ fiind mutual neinterferente.
- Deci $F(M_j\ ,\beta_1) = F(M_j\ ,\beta_1\beta_2\beta_3)$ pentru $\forall M_j \in D_S$, iar S scrie aceeaşi valoare în $\forall\ M_i \in R_S$ atât pentru α cât şi pentru α'.
- Considerând că v este valoarea înscrisă de S în $M_i \in R_S$ pentru α, rezultă că:

$V(M_i,\alpha) = V(M_i\ ,\ \beta_1\ SI\ \beta_2\ SF\ \beta_3)$

$V(M_i,\alpha) = V(M_i\ ,\ \beta_1\ SI\ \beta_2\ SF)$ nu există $S' \in \beta_3$ care scrie în R_S

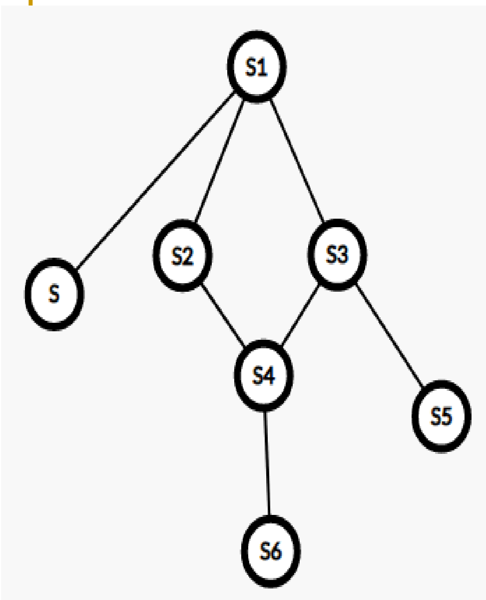
$V(M_i,\alpha) = (V(M_i\ ,\ \beta_1\ SI\ \beta_2)\ ,v)$ S scrie valoarea v în

$V(M_i,\alpha) = (V(M_i\ ,\ \beta_1)\ ,\ v)$ nu există $S' \in \beta_2$ care scrie în R_S

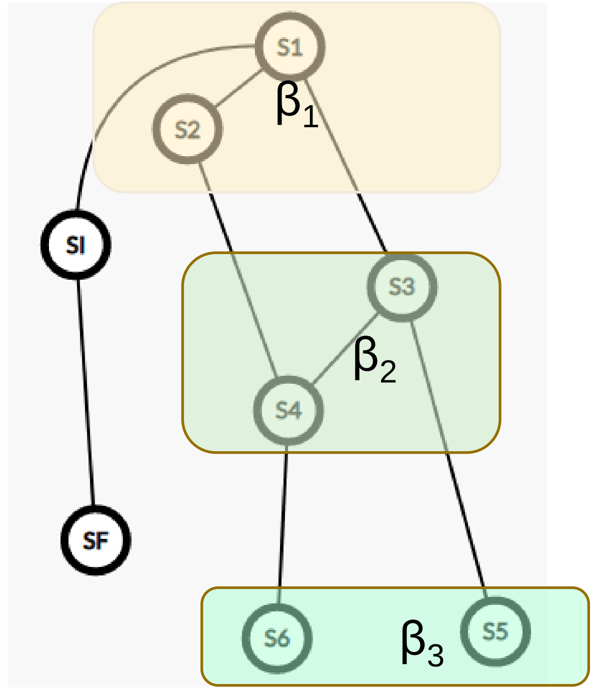
$V(M_i,\alpha) = (V(M_i\ ,\ \beta_1\ \beta_2\ \beta_3),v)$ nu există $S' \in \beta_2\beta_3$ care scrie în R_S

$V(M_i,\alpha) = (V(M_i\ ,\ \beta_1\ \beta_2\ \beta_3\ SI\ SF)$ S scrie v în M_i

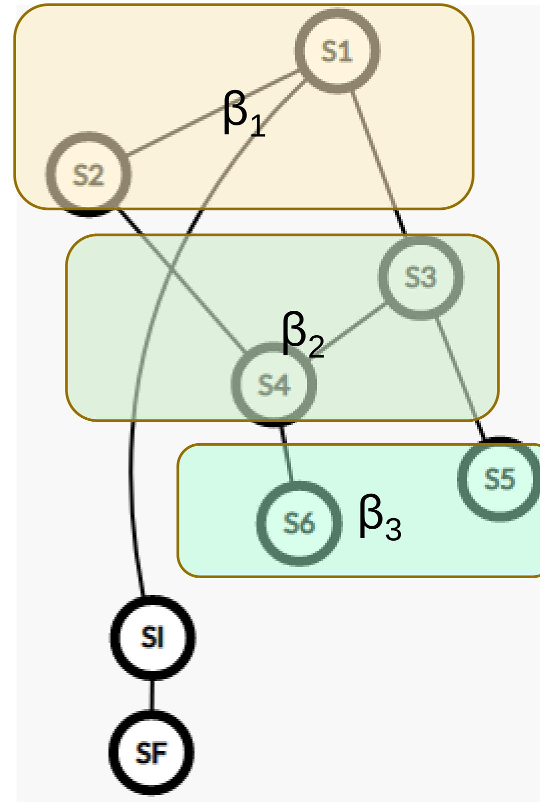
$V(M_i,\alpha) = V(M_i\ ,\ \alpha')$

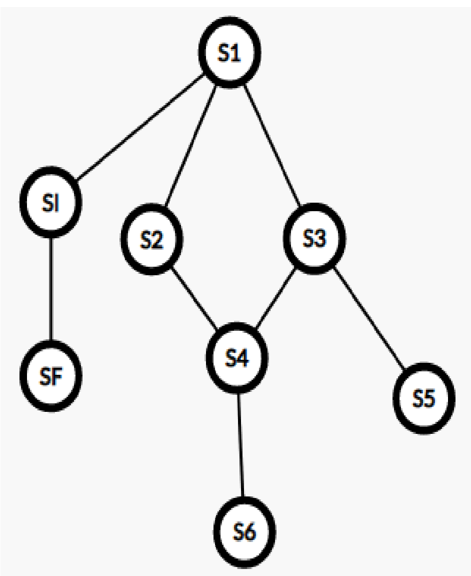


$\alpha = \beta_1 \; SI \; \beta_2 \; SF \; \beta_3$



$\alpha' = \beta_1 \; \beta_2 \; \beta_3 \; SI \; SF$





Lema

Teorema de suficienţă:
Sistemele de sarcini formate din sarcini mutual neinterferente sunt determinate.

Justificare (prin inducţie)

- Pentru un sistem de sarcini format dintr-o singura sarcina este evident.
- Presupunem că afirmaţia este adevărată pentru sisteme cu n-1 sarcini, (α'_1 şi α'_2 sunt secvenţe de execuţie pentru un sistem cu n-1 sarcini)
- Presupunem sistemul **C** = (***S***, **<**) format din n sarcini.

- Dacă sistemul **C** are o singură secvenţă de execuţie validă, el este determinat.
- Presupunem 2 secvenţe de execuţie α_1, α_2 în **C** şi S sarcina terminala în ***S***.

Conform lemei putem forma 2 secvenţe de execuţie α'_1 şi α'_2 astfel:

$$\alpha_1 = \alpha'_1 \ \mathrm{SI}\ \ \mathrm{SF}; \quad \alpha_2 = \alpha'_2\ \mathrm{SI}\ \mathrm{SF},$$

în condiţiile în care

$$V(M_i, \alpha_1) = V(M_i, \alpha'_1), \ \ 1 \le i \le m$$

$$V(M_i, \alpha_2) = V(M_i, \alpha'_2), \ \ 1 \le i \le m$$

α'_1 şi α'_2 sunt secvenţe de execuţie pentru un sistem cu n-1 sarcini:

C' = (***S*** \ S, **<**'), relaţia de precedenţă **<**' fiind obţinută din **<** eliminând relaţia de precedenţă ce implică pe S.

$$V(M_i, \alpha'_1) = V(M_i, \alpha'_2) \text{ din ipoteza de inducţie.}$$

- Deci se poate presupune că valorile din D_S sunt aceleaşi, atât pentru α'_1 cât şi pentru α'_2,
- respectiv $F(M_i, \alpha'_1) = F(M_i, \alpha'_2)$ pentru $M_i \in D_S$.
- Rezultă astfel că pentru α'_1 şi α'_2, sarcina S scrie aceeaşi valoare v pentru orice celulă $M_i \in R_S$.

Pentru $\mathbf{M_i \not\in R_S}$:

$\mathbf{V(M_i, \alpha_1)}$ = $V(Mi , \alpha'1$ SI SF) Lemă

= $V(M_i , \alpha'_1)$ $M_i \not\in R_{S\ si}$ ip. de inducţie

= $V(M_i , \alpha'_2)$ $M_i \not\in R_S$

= $V(M_i , \alpha'_2$ SI SF) Lemă

= $\mathbf{V(M_i , \alpha_2)}$

Pentru $M_i \not\in R_S$ $\mathbf{V(M_i, \alpha_1) = V(M_i, \alpha_2)}$

Pentru $\mathbf{M_i \in R_S}$:

$\mathbf{V(M_i, \alpha_1)}$ = $V(Mi , \alpha'1$ SI SF) Lemă

= $(V(M_i , \alpha'_1),v)$ ip. de inducţie

= $(V(M_i , \alpha'_2),v)$

= $V(M_i , \alpha'_2$ SI SF) Lemă

= $\mathbf{V(M_i , \alpha_2)}$

Pentru Mi ∈ RS **V(Mi, α1) = V(Mi, α2)**

- *Sistemul **C** este determinat dacă sarcinile sunt mutual neinterferente.*

Teorema de necesitate:

Fie **C** un sistem de sarcini astfel că pentru fiecare $S \in \boldsymbol{S}$**,** f_S nu este specificată, dar D_S şi $R_S \neq \Phi$ sunt date.

C este determinat pentru orice interpretare a sarcinilor sale numai dacă acestea sunt mutual neinterferente.

Justificare (reducere la absurd**)**

- Presupunem că S, S' sunt interferente, independente (adică nu există nici o relaţie de ordine între ele).
- Atunci există două secvenţe de execuţie valide:

  α = β1 SI SF SI' SF' β2

  α' = β1 SI' SF' SI SF β2

- Presupunem că există o celulă de memorie $M_i \in R_S \cap R_{S'}$ (sunt considerate interferente) şi putem alege f_S şi $f_{S'}$ (două interpretări, pentru S şi S'), astfel încât pentru:

  f_S: S să scrie în M_i valoarea u;

  $f_{S'}$: S' să scrie în M_i valoarea v; $u \neq v$

În ceea ce priveşte valorile din R_S şi $R_{S'}$ pentru secvenţele α şi α' putem spune că:

$$V(M_i, \alpha) = V(M_i, \beta 1 \text{ SI SF SI' SF'}) = (V(M_i, \beta_1), u, v);$$

$$u \neq v$$

$$V(M_i,\alpha') = V(M_i, \beta_1 \text{ SI' SF' SI SF}) = (V(M_i, \beta_1), v, u);$$

Deci în celula M_i, secvenţele α si α' înscriu secvenţe de valori diferite

$$(V(M_i, \beta_1), u, v); \text{ respectiv } (V(M_i, \beta_1), v, u);$$

$$V(M_i, \alpha) \neq V(M_i, \alpha')$$

ceea ce înseamnă că sistemul de sarcini **C** nu este determinat.

Deci este necesar ca $R_S \cap R_{S'} = \Phi$ pentru că altfel rezultă că **C** este nedeterminat.

- Presupunem că există o celulă $M_j \in D_S \cap R_{S'}$.
- Presupunem că există o celulă $M_i \in R_S$.
- Atunci, putem alege o interpretare a lui S', o funcţie $f_{S'}$ astfel încât $F(M_j, \beta_1) \neq F(M_j, \beta_1$ SI' SF').
- Rezultă în acest caz că S citeşte valori diferite pentru α şi α' (având în vedere că $M_j \in D_S \cap R_{S'}$).

- Putem alege în acest caz o funcţie f_S astfel încât sarcina S să scrie în M_i, valoarea u pentru secvenţa α şi valoarea v pentru secvenţe α', cu $u \neq v$.

În acest caz putem spune că:

$V(M_i, \alpha)$ $= V(M_i, \beta_1$ SI SF SI' SF')

$= V(M_i, \beta_1$ SI SF) $\qquad R_S \cap R_{S'} = \Phi$

$= (V(M_i, \beta_1),u)$ $\qquad$ S scrie u pentru α

$$V(M_i, \alpha') = V(M_i, \beta_1\ SI'\ SF'\ SI\ SF)$$
$$= (V(M_i, \beta_1\ SI'\ SF'),v)$$
$$= (V(M_i, \beta_1),v)$$ S scrie v pentru α'

deci $V(M_i, \alpha) \neq V(M_i, \alpha')$ deci **C** nu este determinat.

Rezultă că trebuie ca : $D_S \cap R_{S'} = \Phi$

Similar se poate arăta că trebuie ca : $D_{S'} \cap R_S = \Phi$

O aplicaţie foarte importantă a teoremelor arătate anterior se referă la paralelismul maxim în sisteme de sarcini în care constrângerile de precedenţă sunt impuse numai de cerinţele de determinare.

- Din definiţia dată pentru determinare rezultă că fiecare sarcina produce o singură secvenţă de valori pentru o celulă de memorie M_i în condiţiile unei stări inţiale date.

- Două sisteme cu aceeaşi mulţime de sarcini S sunt echivalente dacă sunt determinate şi dacă pentru aceeaşi stare iniţială produc aceeaşi secvenţă de valori.

Un sistem de sarcini **C** şi graful asociat G (definit de relatia de precedenta **<**) implică un paralelism maxim dacă **C** este determinat şi dacă **eliminarea** unui arc oarecare (S, S') din G, face ca sarcinile S şi S' să devină **interferente**.

- Astfel dacă (S , S') este un arc într-un graf cu maximum de paralelism atunci:

$$(R_S \cap R_{S'}) \cup (R_S \cap D_{S'}) \cup (D_S \cap R_{S'}) \neq \Phi$$

- Având un sistem de sarcini **C**, determinat, este util să construim sistemul echivalent **C**' cu maximum de paralelism.

Sistem de sarcini cu maximum de paralelism

Teoremă :

Avand un sistem de sarcini **C**= (***S***, **<**) se poate construi un sistem

C' = (***S***, **<**') unde

relaţia <' este închiderea prin tranzitivitate a relaţiei:

$Z=\{(S,S') \in < \mid (R_S \cap R_{S'}) \cup (R_S \cap D_{S'}) \cup (D_S \cap R_{S'}) \neq \Phi\}$

C' este unicul sistem echivalent cu **C** care implică **maximum de paralelism.**

Exemplu :

Fie sistemul de sarcini C = (S , **<**) unde

S = {S1,S2,S3,S4,S5,S6,S7,S8 }

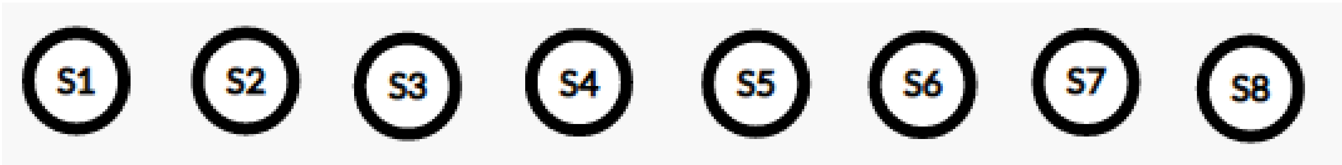
Să se construiască sistemul de sarcini C'=(S, **<**') echivalent, care realizează maxim de paralelism tinand seama de cerinţele de determinare (care are la baza conflictul de resurse)

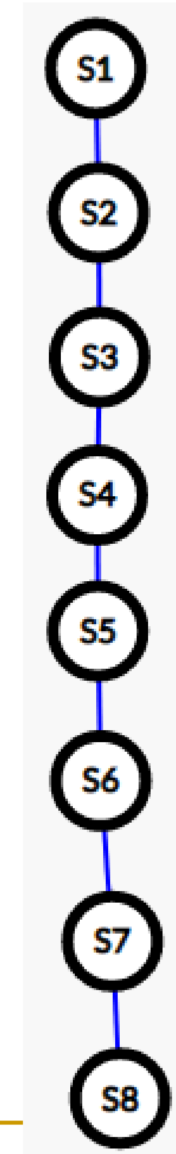
- Sistemul este caracterizat prin:
- $M = \{ \; M_1, M_2, M_3, M_4, M_5 \}$

iar domeniile D_S şi R_S sunt specificate prin tabelul :

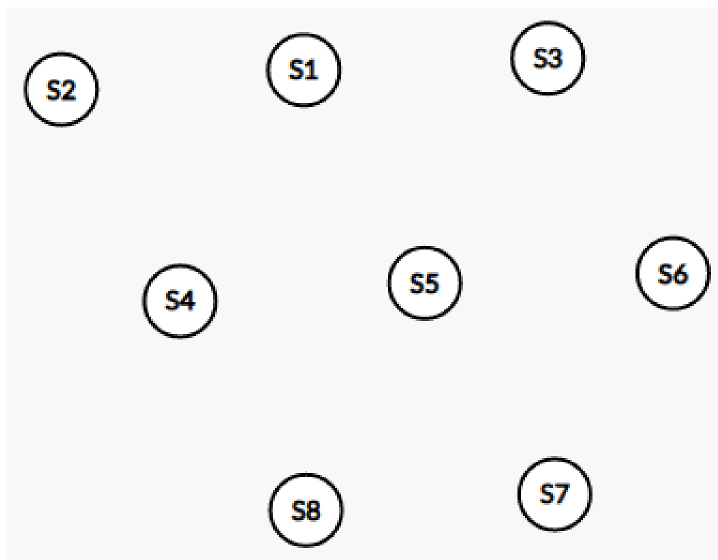
M	Domeniu de definiţie D_S	Domeniu de valori R_S
M1	S1,S2,S7,S8	S3
M2	S1,S7	S5
M3	S3,S4,S8	S1
M4	S3,S4,S5,S7	S2,7
M5	S6	S4,S6,S8

Posibillitati de executie a sistemului de sarcini





maxim de paralelism tinand seama de cerinţele de determinare bazate pe eliminarea conflictului de resurse



$$Z=\{(S,S') \in < \mid (R_S \cap R_{S'}) \cup (R_S \cap D_{S'}) \cup (D_S \cap R_{S'}) \neq \Phi\}$$

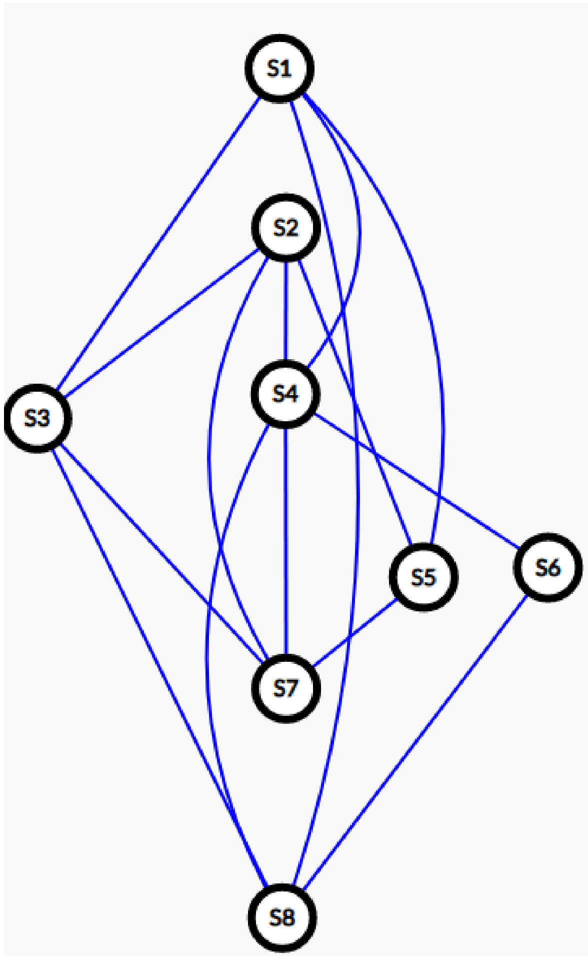
Relaţia Z, conform definiţiei de mai sus va fi dată de mulţimea arcelor:

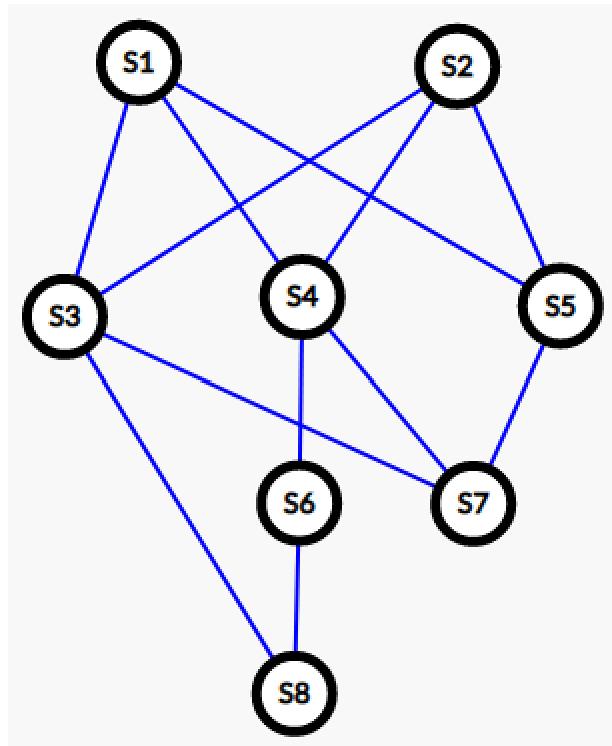
Z : { (S1,S3) (S1,S4) (S1,S5) (S1,S8)
(S2,S3) (S2,S4) (S2,S5) (S2,S7)
(S3,S7) (S3,S8)
(S4,S6) (S4,S7) (S4,S8)
(S5,S7)
(S6,S8)
}

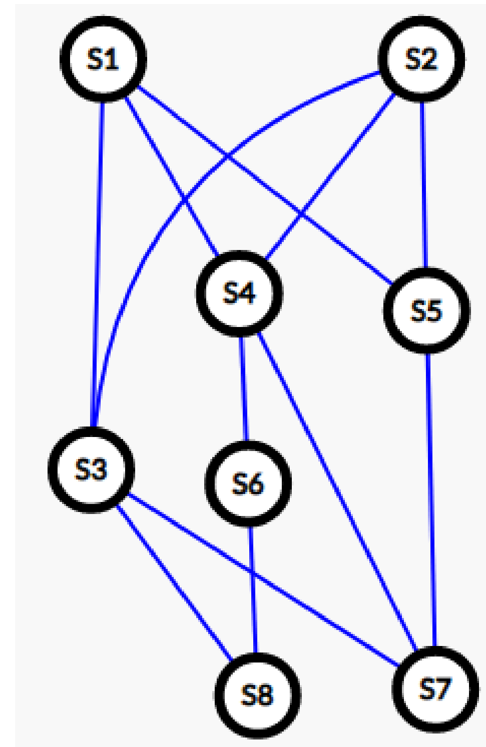
M	Domeniu de definiţie D_S	Domeniu de valori R_S
M1	S1,S2,S7,S8	S3
M2	S1,S7	S5
M3	S3,S4,S8	S1
M4	S3,S4,S5,S7	S2,7
M5	S6	S4,S6,S8

Rezultă graful asociat G’, care este organizat pe niveluri eliminând arcele excluse prin tranzitivitate

graful asociat







Sistemul C' echivalent se execută în 4 perioade de timp considerând că avem un număr suficient de procesoare (3 procesoare) sau in 4 perioade avand la dispozitie numai 2 procesoare