Etapa 2

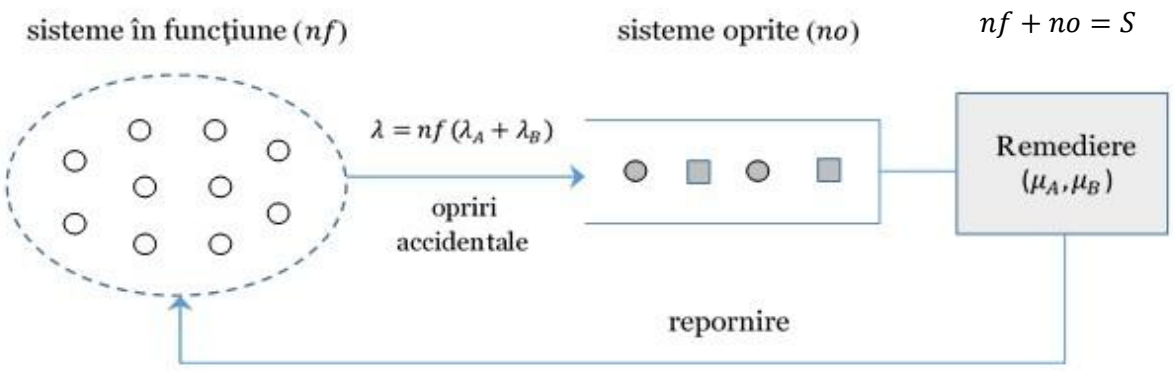
Program de simulare pentru problema de interferență în care un muncitor deservește mai multe sisteme identice

Stati Andreea Grupa:1310A

1. Analiza detaliată a problemei de interferență

- ***Modelarea problemei***

Ca model de simulare, sistemul în ansamblu compus din cele S mașini automate și muncitorul de deservire poate fi privit ca un sistem de servire cu o stație, așa cum este ilustrat în figura următoare.

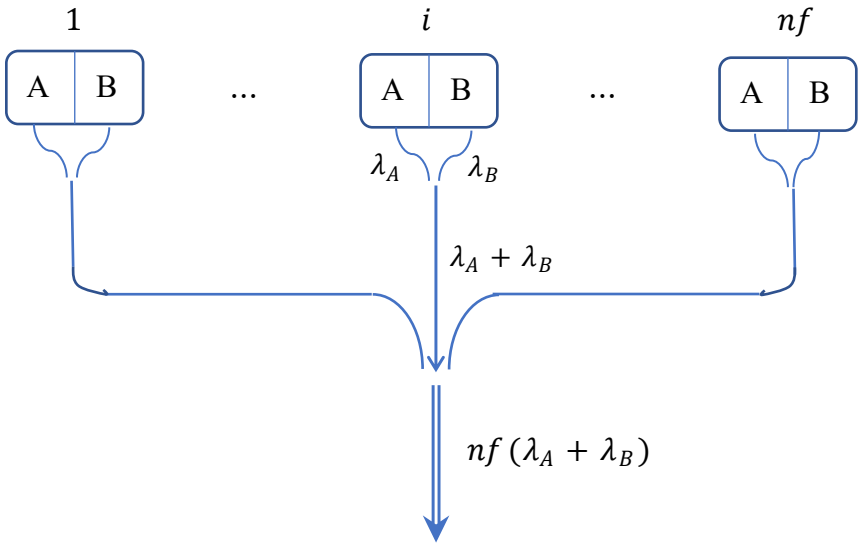


Modelul evidențiază un sistem de servire cu cereri de tip diferit. Principala diferență față de modelele studiate anterior apare însă la fluxul de alimentare cu cereri. Să analizăm în continuare acest aspect.

- ***Fluxul opririlor***

Față de toate sistemele studiate anterior, în acest caz rata medie a cererilor pentru muncitorul de deservire nu mai este constantă în timp întrucât depinde de numărul de sisteme în funcțiune (nf). Rata este maximă atunci când toate sistemele funcționează și devine 0 când toate sistemele sunt oprite. Acest aspect al unui flux de intrare variabil

în timp nu a mai fost studiat până acum. De la prima etapă a rezultat că cele două variabile aleatoare primare T_fA și T_fB au repartiții exponențial negative, de parametru λ_A și respectiv, λ_B. Orice întrerupere accidentală duce la oprirea mașinii pentru remediere. Prin urmare, timpul de funcționare a unei mașini până apare o oprire accidentală este $T_f = min\{T_fA, T_fB\}$. Pe baza proprietății studiate la curs rezultă că și variabila aleatoare T_f are tot o repartiție exponențial negativă de parametru $\lambda_A + \lambda_B$. Cu alte cuvinte, prin suprapunerea efectelor celor două cauze independente de întrerupere accidentală rezultă pentru o mașină un flux al opririlor de tip Poissonian cu o rată medie egală cu $\lambda_A + \lambda_B$. Dar, prin reunirea mai multor fluxuri Poissoniene independente rezultă tot un flux Poissonian. Așadar, atunci când sunt nf sisteme în funcțiune rata medie a opririlor este $\lambda = nf(\lambda_A + \lambda_B)$, iar durata dintre două opriri consecutive are o repartiție exponențial negativă de același parametru λ. Figura următoare ilustrează aspectele menționate anterior.



De remarcat că cele nf mașini nu sunt puse în funcțiune în același timp. Dacă ținem cont de proprietatea variabilei aleatoare exponențial negative T_f de a fi "fără memorie" acest aspect nu mai are relevanță. Prin urmare, rezultă că pentru cele nf mașini în funcțiune durata dintre două opriri consecutive are într-adevăr o repartiție exponențial negativă de parametru $\lambda = nf(\lambda_A + \lambda_B)$. În aceste condiții fluxul opririlor este ușor de simulat apelând doar funcția de generate $genExp(\lambda)$.

- *Deservirea*

Timpul de remediere a unui sistem oprit depinde de tipul modulului afectat de întreruperea accidentală, A sau B. Fie p_A și $p_B = 1 - p_A$ probabilitatea ca la un sistem oprit modulul care necesită remediere să fie de tip A și respectiv, de tip B. Se deduc ușor relațiile de calcul

$$pA = \lambda A/(\lambda A + \lambda B) \text{ și } pB = \lambda B/(\lambda A + \lambda B) .$$

De observat că

$$pA/pB = \lambda A/\lambda B.$$

Timpul mediu de remediere a unui sistem oprit se exprimă cu relația:

$$Tr_m = p_A Tr_m^A + p_B Tr_m^B$$

Dar variabilele aleatoare T_rA și T_rB au repartiții exponențial negative, de parametru μ_A și respectiv, μ_B. Ca urmare, $Tr_m{}^A = 1/\mu_A$ și $Tr_m{}^B = 1/\mu_B$. Pentru timpul mediu de remediere a unui sistem oprit rezultă relația de calcul:

$$Tr_m = p_A Tr_m^A + p_B Tr_m^B = \lambda_A/(\lambda_A + \lambda_B) \cdot 1/\mu_A + \lambda_B/(\lambda_A + \lambda_B) \cdot 1/\mu_B =$$
$$= 1/(\lambda_A + \lambda_B) \cdot (\lambda_A/\mu_A + \lambda_B/\mu_B).$$

Acest rezultat este foarte util la validarea programului de simulare.

Pentru generarea valorilor de selecție privind timpul de remediere, care să reflecte amestecul celor două tipuri de operații care se succed în mod aleatoriu, cu respectarea proporției date de parametrii λA și λB, se procedează așa cum este prezentat în cursul 4, care tratează sistemele de servire cu cereri de tip diferit.

2. Algoritmul de simulare

Fie S numărul de sisteme identice deservite de muncitor. În funcție de valoarea lui S trebuie să se determine disponibilitatea sistemelor și gradul de ocupare a muncitorului de deservire.

Semnificația variabilelor folosite în algoritmul de simulare este următoarea:

- $ceas, DS$ – ceasul și durata simulării
- nf și no – variabile de stare; $nf + no = S$
- Tpo – timpul până la o nouă oprire a unui sistem în funcțiune; variabila nu are semnificație când $nf = 0$.
- Tr – timpul necesar pentru remedierea sistemului în curs; variabila nu are semnificație atunci când $no = 0$.
- STf – statistică cu suma timpilor de funcționare pentru cele S sisteme în perioada de monitorizare.
- STr – statistică cu suma timpilor de remediere.

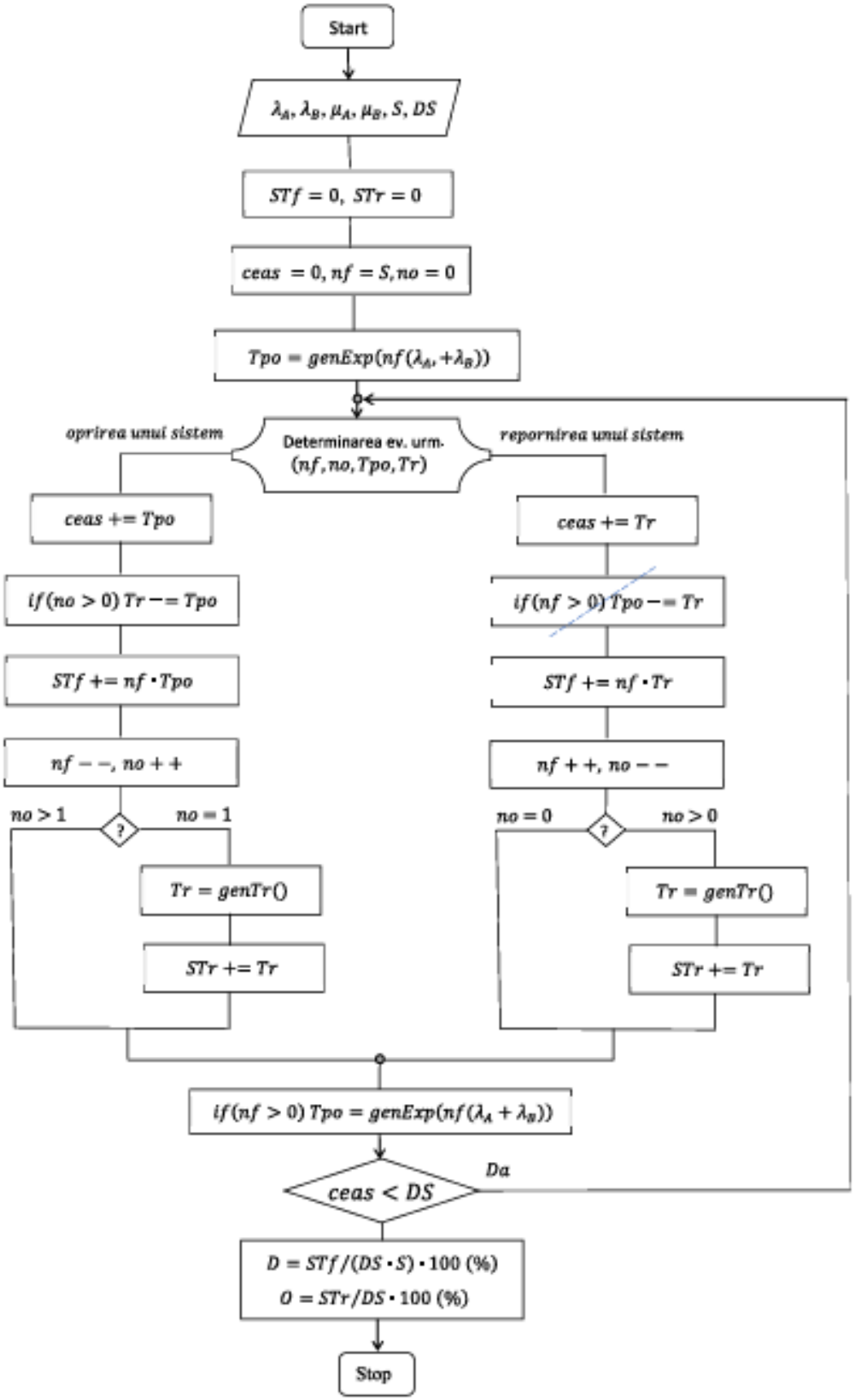
La sfârșitul simulării mărimile de interes se determină cu relațiile:

$$D = STf/(DS \cdot \mathrm{S}) * 100\ (\%)$$

$$O = STr/DS * 100\ (\%)$$

Pentru a putea simula fluxul opririlor în care intervine variabila de stare nf algoritmul de simulare trebuie să urmărească explicit toate schimbările de stare din sistem în perioada simulată. Algoritmul de simulare este prezentat în figura următoare (Fig. 7).

Programul de simulare se rulează pentru valori tot mai mare ale lui S până când gradul de ocupare a muncitorului de deservire depășește 90%.


Start
$\lambda_A, \lambda_B, \mu_A, \mu_B, S, DS$
$STf = 0,\ STr = 0$
$ceas = 0, nf = S, no = 0$
$Tpo = genExp(nf(\lambda_A + \lambda_B))$
oprirea unui sistem
Determinarea ev. urm.
(nf, no, Tpo, Tr)
repornirea unui sistem
$ceas += Tpo$
$if(no > 0)\ Tr -= Tpo$
$STf += nf \cdot Tpo$
$nf --, no ++$
$no > 1$
?
$no = 1$
$Tr = genTr()$
$STr += Tr$
$ceas += Tr$
$if(nf > 0)\ Tpo -= Tr$
$STf += nf \cdot Tr$
$nf ++, no --$
$no = 0$
?
$no > 0$
$Tr = genTr()$
$STr += Tr$
$if(nf > 0)\ Tpo = genExp(nf(\lambda_A + \lambda_B))$
$ceas < DS$
Da
$D = STf/(DS \cdot S) \cdot 100\ (\%)$
$O = STr/DS \cdot 100\ (\%)$
Stop

3. Codul sursă

```
#include <cmath>
#include <iostream>
using namespace std;

double genExp(double lambda) {
	double u, x;
	u = (double)rand() / (RAND_MAX+1);
	x = -1 / lambda * log(1 - u);
	return x;
}

double genGauss(double medie, double sigma)
{
	double s = 0; int i;
	for (i = 1; i <= 12; i++)
		s += (double)rand() / (RAND_MAX + 1);
	return medie + sigma * (s - 6);
}

double genTr(double PA, double mA, double mB, double sigmaA, double
sigmaB) {
	double U, x;
	U = (double)rand() / (RAND_MAX);
	if (U < PA)
		x = genGauss(mA, sigmaA);
	else
		x = genGauss(mB, sigmaB);
	return x;
}

int main() {
	double lambdaA = 0.2023;
	double lambdaB = 0.1917;

	double miuA = 3.6015;
	double miuB = 2.1835;

	double mA = 1.0 / miuA;
	double mB = 1.0 / miuB;

	double sigmaA = (double)1 / (4 * miuA);
	double sigmaB = (double)1 / (4 * miuB);

	double PA = lambdaA / (lambdaA + lambdaB);
	double PB = lambdaB / (lambdaA + lambdaB);
```

```
int S = 1; // nr de sisteme deservite de muncitor
double DS; // durata de simulare

double STf = 0; //  statistica cu suma timpilor de functionare pentru cele S sisteme in perioada de monitorizare.
double STr = 0; //  statistica cu suma timpilor de remediere.

double ceas = 0; // ceasul simularii
int nf = S;
int no = 0;

double Tpo; // timpul pana la o noua oprire a unui sistem in functiune
double Tr; // timpul necesar pentru remedierea sistemului in curs

double D1, D; // disponibiltate;
D1 = (1.0 / (1.0 + lambdaA / miuA + lambdaB / miuB)) * 100; // teoretica
double O = 0; // grad de ocupare

//  timpul mediu de functionare pana la o oprire accidentala
double Tfm = 1.0 / (lambdaA + lambdaB);

double Trm_teoretic, Trm_calculat; // timpul mediu de remediere
Trm_teoretic = 1 / (lambdaA + lambdaB) * (lambdaA / miuA + lambdaB / miuB);   //timpul mediu de remediere

long NO; // nr de opriri care apar in perioada de simulare
long NR;

while (O <= 90) { /* Programul de simulare se ruleaza pentru valori tot mai mare  ale lui S pana cand gradul de ocupare a muncitorului de deservire depaseste 90% */

        STf = 0;
        STr = 0;
        ceas = 0;
        nf = S;
        no = 0;
        NO = 0;
        NR = 0;
        Tpo = genExp(S * (lambdaA + lambdaB));

        while (NO < 1000000) {
```

```
            if ((nf > 0 && nf < S && Tpo < Tr) || (nf == S)) {
                  NO++;
                  ceas += Tpo;

                  if (no > 0)
                        Tr -= Tpo;

                  STf += nf * Tpo;
                  nf--;
                  no++;

                  if (no == 1) {
                        Tr = genTr(PA, mA, mB, sigmaA, sigmaB);
                        STr += Tr;
                  }
            }
            else {
                  NR++;
                  ceas += Tr;
                  STf += nf * Tr;
                  nf++;
                  no--;

                  if (no > 0) {
                        Tr = genTr(PA, mA, mB, sigmaA, sigmaB);
                        STr += Tr;
                  }
            }
            if (nf > 0)
                  Tpo = genExp(nf * (lambdaA + lambdaB));
            DS = ceas;
      }

      D = (STf / (DS * S)) * 100;
      O = STr / DS * 100;
      Trm_calculat = STr / NR;
      if (S == 1) {
            cout << "Disponibilitate analitica = " << D1 <<
"\n\n";
      }

      cout << "\nNr sisteme=" << S << endl;
```

```
            cout << "Disponibilitate  D=" << D << endl;
            cout << "Grad ocupare  O=" << O << endl;

            if (S == 1)
                  cout << "D+O=" << D + O << endl;

            cout << "Nr. opriri=" << NO << endl;
            cout << "Perioada totala de functionare a tuturor celor "
<< S << " sisteme: " << DS * S * (D / 100) * (lambdaA + lambdaB) <<
endl;
            cout << "Nr. remediere=" << NR << endl;
            cout << "no=" << no << endl;
            cout << "Trm teoretic=" << Trm_teoretic << endl;
            cout << "Trm calculat=" << Trm_calculat << endl;
            S++;
      }
      return 0;
}
```

4. Rezultate obținute

S	1	2	3	4	5	6	7	8
$D(\%)$	87.4302	86.5651	85.523	84.2616	82.6064	80.4861	77.8928	74.7003
O (%)	12.5698	24.9436	36.9985	48.4797	59.4849	69.6356	78.5995	86.0496

Trm teoretic=0.365396

Disponibilitate analitica = 87.4152

Nr sisteme: S=1

Disponibilitate: D=87.4302

Grad ocupare: O=12.5698

D+O=100

Nr. opriri: NO=1000000

Perioada totala de functionare a tuturor celor 1 sisteme: 1.00112e+06

Nr. remediere: NR=999999

no=1

Trm teoretic=0.365396

Trm calculat=0.365307

Nr sisteme: S=2

Disponibilitate: D=86.5651

Grad ocupare: O=24.9436

Nr. opriri: NO=1000000

Perioada totala de functionare a tuturor celor 2 sisteme: 1.00019e+06

Nr. remediere: NR=999998

no=2

Trm teoretic=0.365396

Trm calculat=0.365741

Nr sisteme: S=3

Disponibilitate: D=85.523

Grad ocupare: O=36.9985

Nr. opriri: NO=1000000

Perioada totala de funcționare a tuturor celor 3 sisteme: 998835

Nr. remediere: NR=999999

no=1

Trm teoretic=0.365396

Trm calculat=0.365577

Nr sisteme: S=4

Disponibilitate: D=84.2616

Grad ocupare: O=48.4797

Nr. opriri: NO=1000000

Perioada totala de functionare a tuturor celor 4 sisteme: 1.00102e+06

Nr. remediere: NR=999999

no=1

Trm teoretic=0.365396

Trm calculat=0.365443

Nr sisteme: S=5

Disponibilitate: D=82.6064

Grad ocupare: O=59.4849

Nr. opriri: NO=1000000

Perioada totala de functionare a tuturor celor 5 sisteme: 999449

Nr. remediere: NR=999997

no=3

Trm teoretic=0.365396

Trm calculat=0.365333

Nr sisteme: S=6

Disponibilitate: D=80.4861

Grad ocupare: O=69.6356

Nr. opriri: NO=1000000

Perioada totala de functionare a tuturor celor 6 sisteme: 999094

Nr. remediere: NR=999998

no=2

Trm teoretic=0.365396

Trm calculat=0.365654

Nr sisteme: S=7

Disponibilitate: D=77.8928

Grad ocupare: O=78.5995

Nr. opriri: NO=1000000

Perioada totala de functionare a tuturor celor 7 sisteme: 999575

Nr. remediere: NR=999998

no=2

Trm teoretic=0.365396

Trm calculat=0.365716

Nr sisteme: S=8

Disponibilitate: D=74.7003

Grad ocupare: O=86.0496

Nr. opriri: NO=1000000

Perioada totala de functionare a tuturor celor 8 sisteme: 1.00004e+06

Nr. remediere: NR=999997

no=3

Trm teoretic=0.365396

Trm calculat=0.365478

Nr sisteme: S=9

Disponibilitate: D=70.7959

Grad ocupare: O=91.7544

Nr. opriri: NO=1000000

Perioada totala de functionare a tuturor celor 9 sisteme: 999932

Nr. remediere: NR=999993

no=7

Trm teoretic=0.365396

Trm calculat=0.365472

5. Verificări pentru asigurarea corectitudinii programului de simulare

Verificări preliminare:

Compararea numărului de opriri cu suma dintre numărul de remedieri si numărul de sisteme rămase oprite la sfârșitul simulării;

Compararea raportului STr/Nr cu valoarea teoretică;

$$\frac{STr}{Nr} = \frac{1}{(\lambda_A + \lambda_B)}\left(\frac{\lambda_A}{\mu A} + \frac{\lambda_B}{\mu B}\right)$$

Verificarea rezultatelor simulării (Disponibilitatea și Gradul de ocupare):

Pentru un singur sistem, gradul de disponibilitate poate fi caracterizat cu formula:

$$\boldsymbol{D} = \frac{1}{1 + \frac{\lambda_A}{\mu A} + \frac{\lambda_B}{\mu B}} \cdot 100(\%)$$

Pentru mai multe sisteme se va face o verificare cantitativă asupra disponibilității sistemului. De asemenea, gradul de ocupare al muncitorului ar trebui să crească aproximativ liniar cu numărul de sisteme.

$$No = DS \cdot s \cdot \frac{D}{100} \cdot (\lambda_A + \lambda_B)$$

6. Concluzii

Observăm că pe măsură ce numărul de sisteme alocate muncitorului crește, gradul său de ocupare înregistrează o creștere, generând în consecință o reducere a gradului de disponibilitate a sistemului.