

Universitatea din București
Facultatea de Matematică și Informatică

CURS nr. 8 – TEHNICI DE SIMULARE

Modele de simulare

Lect. dr. Bianca Mogoș

Conținut

1. Principiile modelării și simulării
 - 1.1 Noțiuni de bază – Definiție model
 - 1.2 Introducerea noțiunii de simulare
 - 1.3 Avantajele și dezavantajele utilizării simulării
 - 1.4 Aplicații ale simulării
 - 1.5 Schema de modelare și simulare
 - 1.6 Construcția unui model de simulare.
 - 1.7 Exemplu: Modelul de simulare asociat unui sistem de așteptare
2. Generalități despre limbajul GPSS
 - 2.1 Descriere generală a limbajului GPSS
 - 2.2 Structura instrucțiunii GPSS
 - 2.3 Blocuri de creare și distrugere de tranzacții
 - 2.4 Blocuri de acțiune
 - 2.5 Blocuri pentru obținerea de statistici
 - 2.6 Blocuri de control logic al tranzacțiilor
 - 2.7 Blocuri de modificare a caracteristicilor tranzacțiilor și a valorilor unor entități de referință

1.1 Noțiuni de bază – Definiție model (1)

- ▶ Oamenii de știință și inginerii încearcă *să înțeleagă, să dezvolte și să optimizeze sisteme.*
- ▶ O *problemă* care apare în analiza unui sistem este *complexitatea* acestuia.
- ▶ Folosirea unor *descrieri simplificate* ale sistemelor – *modelele* – reprezintă o *soluție* care permite abordarea unor sisteme complexe.
- ▶ *Definiție:* Pentru un observator B , un obiect A^* este un *model* al obiectului A dacă B poate folosi A^* pentru a răspunde întrebărilor care îl interesează referitoare la A .
- ▶ *Modelul cel mai bun* este *modelul cel mai simplu* care își îndeplinește obiectivul de a ne ajuta să înțelegem un sistem complex și de a rezolva anumite probleme.

1.2 Introducerea noțiunii de simulare

- ▶ Cuvântul *simulare* este de origine latină și înseamnă *capacitatea de a reproduce ceva*.
- ▶ Termenul de simulare a fost introdus de *John von Neumann* la începutul anilor '40.
- ▶ *Modelul de simulare* este o extindere a unui model matematic prin descrierea relațiilor dintre componentele modelului.
- ▶ *Simularea* reprezintă aplicarea unui model (numit model de simulare) pentru a descrie comportamentul și evoluția unui sistem real în timp.

1.3 Avantajele și dezavantajele utilizării simulării (1)

Avantajele obținute datorită realizării unui studiu de simulare

- ▶ Simularea furnizează o modalitate *necostisitoare* și *neriscantă* de realizare a unor teste referitoare la funcționalitatea unui sistem.
- ▶ Simularea permite *prezicerea* comportamentului unui sistem pentru diferite ipoteze/variabile de intrare ale sistemului și *reducerea riscului* de a lua o decizie greșită.
- ▶ Un model de simulare permite *analiza variabilității proceselor* dintr-un sistem, ne poate ajuta să înțelegem cum interacționează între ele mai multe componente ale sistemului și cum poate afecta interacțiunea dintre ele performanța sistemului.
- ▶ Simularea dă *posibilitatea de explorare* a mai multor căi alternative.
- ▶ Dezvoltarea unei aplicații de simulare poate determina *o mai bună înțelegere* a sistemului studiat.

1.3 Avantajele și dezavantajele utilizării simulării (2)

Dezavantajele unui studiu de simulare

- ▶ Simularea folosește un model – reprezentând o descriere simplificată a sistemului real – care nu poate surprinde întreaga complexitate a unui sistem real.

1.4 Aplicații ale simulării

- ▶ Simularea unui aeroport având ca scop minimizarea întârzierilor provocate de aglomerări
- ▶ Simularea spațiului aerian având ca scop utilizarea optimă a acestuia
- ▶ Simularea în domeniul medical a numărului de cadre medicale necesar la momente diferite de timp
- ▶ Planificarea strategică a operațiilor realizate într-o bancă și realizarea unei topologii optime a sucursalelor băncii
- ▶ Simularea unei linii de producție pentru a estima cantitatea de materie primă necesară și pentru a prezice profitul obținut

1.5 Schema de modelare și simulare

Definiții

- ▶ Definirea problemei pe care ne propunem să o rezolvăm sau a întrebării la care trebuie să răspundem
- ▶ Definirea sistemului – o colecție de obiecte/entități ale căror proprietăți vrem să le studiem

Analiza sistemului

- ▶ Identificarea părților sistemului relevante pentru problemă

Modelarea

- ▶ Dezvoltarea modelului asociat sistemului

Simularea

- ▶ Aplicarea modelului în studiul problemei
- ▶ Dezvoltarea unei strategii pentru rezolvarea problemei

Validarea

- ▶ Verificarea dacă strategia elaborată în pasul de simulare rezolvă problema unui sistem real

1.6 Construcția unui model de simulare (1) – Ceasul simulării

- ▶ *Ceasul simulării*: asigură *ordonarea* corectă în timp a *evenimentelor* create de model și uneori ajută la implementarea condiției de terminare a simulării.
- ▶ *Evenimentele* corespund unor modificări în sistem; de exemplu modificarea valorilor unor variabile care se calculează sau se generează prin instrucțiuni ale modelului
- ▶ Ceasul pornește cu valoarea zero la începutul simulării; simularea se realizează până când ceasul atinge o valoare dată inițial T_{max}
- ▶ O *clasificare* a ceasului simulării
 - ▶ *ceas cu creștere constantă*: ceasul crește cu o unitate de timp c constantă, apoi se prelucrează toate evenimentele apărute pe intervalul de timp de lungime c
 - ▶ *ceas cu creștere variabilă*: ceasul crește cu valoarea ce corespunde apariției primului eveniment următor, apoi programul prelucrează evenimentul și se reia ciclul simulării

1.6 Construcția unui model de simulare (2) – Agenda simulării

- ▶ *Agenda* conține elementele/variabilele asociate evenimentelor memorate de modelul de simulare
- ▶ Agenda *A* se compune din două părți:
 - ▶ *AEC – agenda evenimentelor curente* (care au timpul de apariție egal cu valoarea ceasului)
 - ▶ *AEV – agenda evenimentelor viitoare* (care au timpul de apariție ulterior valorii curente a ceasului)
- ▶ Algoritmul de simulare prelucrează numai evenimentele din AEC; prelucrarea unui eveniment înseamnă fie determinarea apariției unui nou eveniment (ce se memorează în AEV) sau modificarea unei stări, fie distrugerea unui eveniment (ștergerea) lui din agendă.

1.6 Construcția unui model de simulare (3) – Algoritmul simulării

- ▶ *Algoritmul simulării* actualizează agenda prin *interacțiunea acesteia cu ceasul*
- ▶ Într-un *ciclu al simulării* ceasul este actualizat, se selectează din agenda *A* evenimentele care fac parte din AEC și se prelucrează aceste evenimente până când AEC devine vidă.
- ▶ Atunci ceasul este crescut din nou și se reia *ciclul de simulare*.

1.7 Exemplu: Modelul de simulare asociat unui sistem de așteptare (1)

- ▶ *Sistemul de așteptare* este o parte a lumii reale în care se produc *aglomerări*.
- ▶ *Entitățile componente ale sistemului*:
 - ▶ *clienții* care sosesc în sistem
 - ▶ *stațiile de servire* (care servesc după anumite reguli clienții)
- ▶ Dacă sosesc mulți clienți și stațiile de servire nu pot să îi servească se formează *cozi de așteptare* în sistem.
- ▶ Administratorul sistemului trebuie să construiască sistemul a.î. nici clienții să nu *aștepte* mult până primesc serviciul, dar nici stațiile să nu *lenevească*, deoarece ar putea produce pierderi proprietarului.
- ▶ *Aplicații ale sistemelor de așteptare*: în comerț, în managementul sistemelor de comunicații și de transport, dar și în dirijarea și funcționarea în timp real a rețelelor de calculatoare.

1.7 Exemplu: Modelul de simulare asociat unui sistem de așteptare (2)

Un model de așteptare conține de obicei următoarele elemente:

- ▶ *Variabile de intrare (cunoscute, de obicei aleatoare):*
 - ▶ TI : timp de intersosire al clienților
 - ▶ TS : timp de servire al unui client
- ▶ *Variabile de ieșire (necunoscute):*
 - ▶ TA : timp de așteptare (sau LC : lungimea cozii)
 - ▶ TL : timp de lenevire (sau NSL : numărul stațiilor care lenevesc)

Scopul modelului: cunoscând repartițiile de probabilitate ale variabilelor TI și TS vrem să obținem informații despre TA și TL și să se stabilească cum trebuie să se realizeze TS a.î. să se optimizeze o anumită funcție criteriu.

2.1 Descriere generală a limbajului GPSS

- ▶ *Limbajele de simulare* sunt limbaje de programare care implementează elementele de bază ale simulării: *ceasul simulării* și *gestionarea memoriei*.
- ▶ *Utilizatorul* se ocupă de *descrierea evenimentelor* și prelucrarea lor.
- ▶ Exemplu de limbaj de simulare: *GPSS (General Purpose System Simulator)*
- ▶ *Entități ale modelului*:
 - ▶ *Blocuri*: entități care descriu activități
 - ▶ *Tranzacții*: sunt elemente create printr-un bloc GENERATE care parcurg blocurile modelului/sistemului secvențial și în funcție de acțiunea blocurilor accesate.
 - ▶ *Stații de servire* sau *facilități*: resurse care oferă un singur serviciu (adică unei singure tranzacții)
 - ▶ *Multistații de serviciu* sau *depozite*: resurse care oferă servicii mai multor tranzacții simultan
 - ▶ *Entități de calcul*: variabile, funcții
 - ▶ *Entități statistice*: cozi, histogramme

2.2 Structura instrucțiunii GPSS

Etichetă	Numele blocului (Cuvânt cheie)	Parametri (separați prin virgulă)
Opțională (pt. referiri)	Verb imperativ indică acțiunea blocului	A,B,C,...

2.3 Blocuri de creare și distrugere de tranzacții

- ▶ *GENERATE* – creează tranzacții. Are forma generală:

GENERATE A,B,C,D,E

- ▶ A – intervalul mediu între sosiri
- ▶ B – abaterea; lungimea intervalului de timp între generarea a două tranzacții consecutive este o valoare întreagă a unei variabile aleatoare uniform distribuite pe intervalul $[A - B, A + B]$
- ▶ C – momentul creării primei tranzacții
- ▶ D – numărul maxim de tranzacții generate
- ▶ E – prioritatea (valoare implicită = 0)

- ▶ *TERMINATE* – determină distrugerea unei tranzacții. Are forma generală:

TERMINATE A

- ▶ A – decrementează parametrul dat la START (în fereastra care apare în urma rulării programului). De exemplu dacă avem "START *nr*" atunci, în momentul în care o tranzacție accesează blocul "TERMINATE A", "*nr*" devine " $nr = nr - A$ ". Programul rulează cât timp " $nr > 0$ ".

2.4 Blocuri de acțiune (1)

- ▶ *SEIZE/RELEASE* – se folosesc împreună și determină ocuparea/eliberarea unei resurse denumită *facilitate*; de asemenea creează o facilitate, numele facilității fiind dat ca parametru. Au forma generală:

SEIZE resursă

.....

RELEASE resursă

- ▶ resursă – este numele facilității

- ▶ *ADVANCE* – determină oprirea/blocarea tranzacției care accesează acest bloc un interval de timp dat de parametri. Are forma generală:

ADVANCE A,B

- ▶ A – timp mediu de blocare/întârziere a tranzacției
- ▶ B – abaterea

2.4 Blocuri de acțiune (2.1)

- ▶ *ENTER/LEAVE* – se folosesc împreună și determină ocuparea/eliberarea unei/mai multor componente dintr-o resursă denumită *punct de servire cu mai multe stații* sau *entitate de depozitare*.
- ▶ *Capacitatea* (numărul de componente care pot fi ocupate) *resursei* și *entitatea asociată* sunt definite prin blocul *STORAGE*.
- ▶ *Forma generală a blocurilor* este descrisă în slide-ul următor.

2.4 Blocuri de acțiune (2.2)

- Sintaxa blocurilor STORAGE/ENTER/LEAVE:

resursă STORAGE capacitate

.....

GENERATE A1,B1

.....

ENTER resursă,spații_ocupate

.....

LEAVE resursă,spații_eliberate

.....

TERMINATE A2

- resursă – este numele punctului de servire cu mai multe stații sau a entității de depozitare
- spații_ocupate – numărul de componente ocupate din entitatea “resursă” când o tranzacție accesează blocul ENTER
- spații_eliberate – numărul de componente eliberate din entitatea “resursă” când o tranzacție accesează blocul LEAVE
- spații_ocupate, spații_eliberate – trebuie să ia valori mai mici sau egale decât valoarea parametrului “capacitate”

2.4 Blocuri de acțiune (3)

- ▶ *PREEMPT/RETURN* – se folosesc împreună și determină ocuparea/eliberarea unei facilități; ocuparea facilității are loc în funcție de prioritatea asociată tranzacției. Au forma generală:
PREEMPT resursă,prioritate,blocTransfer

.....

RETURN resursă

- ▶ resursă – este numele facilității
- ▶ prioritate – poate fi PR sau poate să lipsească.
 - ▶ Dacă este PR atunci o tranzacție care accesează un bloc PREEMPT poate îndepărta tranzacția care ocupă deja facilitatea (pe care a ocupat-o printr-un bloc SEIZE sau PREEMPT) în funcție de prioritate (tranzacția având prioritatea mai mare va ocupa în continuare facilitatea).
 - ▶ Dacă PR lipsește atunci PREEMPT poate îndepărta doar o tranzacție care a ocupat facilitatea prin blocul SEIZE, în caz contrar tranzacția va fi introdusă în coada de așteptare.
- ▶ blocTransfer – este opțional și reprezintă blocul la care este dirijată tranzacția care ocupa inițial facilitatea. Dacă lipsește atunci tranzacția va fi introdusă în coada de așteptare și după eliberarea facilității va primi în continuare serviciul.

2.5 Blocuri pentru obținerea de statistici (1)

- ▶ *QUEUE/DEPART* – se folosesc împreună și permit obținerea de statistici/informații referitoare la coada de așteptare. Au forma generală:

QUEUE coada

.....

DEPART coada

- ▶ coada – este numele cozii de așteptare
- ▶ *ATENȚIE:* blocurile *QUEUE/DEPART* NU determină crearea unei cozi de așteptare, permit doar obținerea de informații referitoare la coada de așteptare, care este creată implicit de către limbajul GPSS în momentul în care o tranzacție accesează unul dintre blocurile de acțiune *SEIZE, ENTER sau PREEMPT*

2.5 Blocuri pentru obținerea de statistici (2.1)

- ▶ *TABLE/TABULATE* – reprezintă grafic histograma asociată unei mulțimi de selecție. Au forma generală:

```
etHist TABLE A,B,C,D
```

```
.....
```

```
GENERATE A1,B1
```

```
.....
```

```
TABULATE etHist
```

```
.....
```

```
TERMINATE A2
```

- ▶ etHist – numele histogramei, parametru obligatoriu
- ▶ A – expresie care furnizează valori de selecție (generate cf. unei repartiții de probabilitate) asupra unei v.a. X
- ▶ B – a_1 : marginea superioară a primului interval de frecvențe: $I_1 = [a_0, a_1]$ (în notațiile din Curs 7)
- ▶ C – $l = a_i - a_{i-1}$: lungimea unui interval de frecvențe
- ▶ D – k : numărul intervalelor de frecvențe

2.5 Blocuri pentru obținerea de statistici (2.2)

Construcția histogramei este iterativă:

- ▶ în momentul în care o tranzacție accesează blocul “TABULATE etHist”, se calculează expresia definită prin parametrul “A” al histogramei cu numele “etHist”;
- ▶ valoarea expresiei reprezintă o valoare de selecție, notată, x_i pentru tranzacția i , asupra unei variabile aleatoare;
- ▶ la fiecare accesare a blocului “TABULATE” de către o tranzacție se actualizează mulțimea valorilor de selecție, $S_n = S_{n-1} \cup \{x_n\}$, unde $S_{n-1} = \{x_1, x_2, \dots, x_{n-1}\}$ este mulțimea de valori de selecție obținută pentru primele $n - 1$ tranzacții, iar x_n este valoarea expresiei date prin parametrul A calculată pentru tranzacția a n – a care accesează blocul “TABULATE”;
- ▶ histograma este actualizată, fiind construită pentru mulțimea valorilor de selecție S_n .

2.5 Blocuri pentru obținerea de statistici (3)

- ▶ *QTABLE/QUEUE/DEPART* – reprezintă grafic histograma asociată timpilor de așteptare la un serviciu. Au forma generală:
etHist QTABLE numeCoada,B,C,D

.....

GENERATE A1,B1

.....

QUEUE numeCoada

.....

DEPART numeCoada

.....

TERMINATE A2

- ▶ etHist – numele histogramei, element obligatoriu
- ▶ numeCoada – numele cozii de așteptare în care sunt introduse tranzacțiile pentru care se calculează timpul de așteptare pentru a primi un serviciu
- ▶ B, C, D – parametri care definesc histograma; au aceleași semnificații ca pentru "TABLE"

2.6 Blocuri de control logic al tranzacțiilor (1)

- ▶ *TEST* – permite compararea unor valori și dirijarea tranzacțiilor în modelul de simulare. Are forma generală:

TEST O A,B,etTransfer

.....

etTransfer BLOC parametri

- ▶ O – este un operator relațional. Pentru un test cu succes trebuie să se verifice relația “A O B”; unde “O” poate lua valorile E = “=”, G = “>”, GE = “≥”, L = “<”, LE = “≤”, NE = “≠”
 - ▶ A, B – sunt valorile comparate
 - ▶ etTransfer – reprezintă eticheta blocului la care este dirijată tranzacția în cazul în care testul nu are succes
- ▶ *Observație:* dacă testul are succes (adică, condiția este adevărată) tranzacția accesează următorul bloc din program.

2.6 Blocuri de control logic al tranzacțiilor (2)

- ▶ *TRANSFER* – permite dirijarea tranzacțiilor în modelul de simulare. Două modalități de utilizare sunt prezentate în cele ce urmează:

TRANSFER ,etTransfer

sau

TRANSFER prob,,etTransfer

unde

etTransfer BLOC parametri

- ▶ etTransfer – este eticheta blocului la care este dirijată tranzacția
- ▶ prob – reprezintă probabilitatea cu care o tranzacție care accesează blocul “TRANSFER” este dirijată la blocul cu eticheta “etTransfer”; cu probabilitatea “1 - prob” tranzacția este trimisă la blocul următor din program

2.7 Blocuri de modificare a caracteristicilor tranzacțiilor și a valorilor unor entități de referință (1)

- ▶ *ASSIGN* – permite atribuirea/modificarea unei valori/valorii unui parametru asociat tranzacției care accesează blocul. Are forma generală:

ASSIGN param[±],valoare

- ▶ param – este identificatorul asociat parametrului/caracteristicii tranzacției; valoarea parametrului poate fi obținută folosind atributul P\$param (dacă “param” este șir de caractere) sau Pparam (dacă “param” este număr)
- ▶ valoare – este valoarea atribuită sau cu care se incrementează sau decrementează parametrul tranzacției
- ▶ [±] – parantezele pătrate nu fac parte din sintaxa blocului; indică opționalitatea parametrilor “+” sau “-”.
 - ▶ Dacă operatorii “+” sau “-” lipsesc atunci P\$param = valoare
 - ▶ Dacă blocul *ASSIGN* are sintaxa: “*ASSIGN* param+,valoare” atunci P\$param = P\$param + valoare
 - ▶ Dacă blocul *ASSIGN* are sintaxa: “*ASSIGN* param-,valoare” atunci P\$param = P\$param - valoare





2.7 Blocuri de modificare a caracteristicilor tranzacțiilor și a valorilor unor entități de referință (2)

- ▶ *SAVEVALUE* – permite atribuirea/modificarea unei valori/valorii unei variabile (globale) atunci când o tranzacție accesează blocul. Are forma generală:

SAVEVALUE var[±],valoare

- ▶ var – este identificatorul asociat variabilei; valoarea variabilei poate fi obținută folosind atributul X\$var (dacă “var” este șir de caractere) sau Xvar (dacă “var” este număr)
- ▶ valoare – este valoarea atribuită sau cu care se incrementează sau decrementează variabila
- ▶ [±] – parantezele pătrate nu fac parte din sintaxa blocului; indică opționalitatea parametrilor “+” sau “-”.
 - ▶ Dacă operatorii “+” sau “-” lipsesc atunci X\$var = valoare
 - ▶ Dacă blocul *SAVEVALUE* are sintaxa: “*SAVEVALUE* var+,valoare” atunci X\$var = X\$var + valoare
 - ▶ Dacă blocul *SAVEVALUE* are sintaxa: “*SAVEVALUE* var-,valoare” atunci X\$var = X\$var - valoare

Bibliografie I

-  Application Areas, site <http://www.simio.com/applications/>,
Ultima accesare: decembrie 2015
-  T. Schreiber, *An Introduction to Simulation Using GPSS/H*,
Disponibil la: , Ultima accesare: decembrie 2015
-  I. Văduva (2004), *Modele de simulare: note de curs*, Editura
Universității din București, București
-  K. Velten (2009), *Mathematical Modeling and Simulation:
Introduction for Scientists and Engineers*, WILEY-VCH Verlag
GmbH & Co. KGaA, Weinheim