

**ТЕХНИЧЕСКИ УНИВЕРСИТЕТ – СОФИЯ**

**ФАКУЛТЕТ „КОМПЮТЪРНИ СИСТЕМИ И ТЕХНОЛОГИИ”**



## **КУРСОВА РАБОТА**

**по „Компютърно моделиране и симулации“**

**Тема 5: Моделиране на център за обработка на заявки с два компютъра**

**Изготвил:**

Кирил Вълков, фак. № 121222194

Специалност: Компютърно и софтуерно инженерство

**Научен ръководител:**

доц. д-р инж. Валентин Христов

София, 2026

# СЪДЪРЖАНИЕ

<b>I. Постановка на задачата</b> . . . . .	<b>1</b>
<b>II. Теоретична част</b> . . . . .	<b>1</b>
2.1. Първа система (Компютър 1) . . . . .	1
2.2. Втора система (Компютър 2) . . . . .	1
<b>III. Имплементация</b> . . . . .	<b>2</b>
3.1. Основни параметри . . . . .	2
3.2. Генериране на заявки . . . . .	2
3.3. Реализация на GPSS . . . . .	3
3.4. Резултати от GPSS симулация . . . . .	3
<b>IV. Резултати от симулацията</b> . . . . .	<b>4</b>
4.1. Статистически данни . . . . .	4
4.2. Сравнение между Python и GPSS . . . . .	4
4.3. Графична визуализация . . . . .	5
<b>V. Заключение</b> . . . . .	<b>6</b>

## I. ПОСТАНОВКА НА ЗАДАЧАТА

Центърът за обработка на заявки се състои от два компютъра.

- Заявките към първия компютър се получават в съответствие с експоненциално разпределение, като средното време между пристигането на заявките е 1 секунда.
- Вторият компютър получава заявки в съответствие с нормално разпределение, като базовото време за обработка е 3 секунди, с максимално отклонение от 1 секунда.
- Обработката на заявка от всеки компютър продължава  $(2 \pm 1)$  секунди.

Необходимо е да се моделира системата за период от половин час (1800 секунди).

## II. ТЕОРЕТИЧНА ЧАСТ

Системата може да се разглежда като съвкупност от две независими системи за масово обслужване (СМО).

### 2.1. Първа система (Компютър 1)

Входящият поток е Пуасонов (интервалите между заявките са експоненциално разпределени) със средна стойност  $\lambda^{-1} = 1$  s. Времето за обслужване е равномерно разпределено в интервала  $[1, 3]$  s (тъй като  $2 \pm 1$  s). Това е система от тип M/G/1.

Тъй като средното време за обслужване е  $E[S] = 2$  s, а средното време между заявките е  $E[A] = 1$  s, коефициентът на натоварване е  $\rho = \frac{E[S]}{E[A]} = 2$ . Тъй като  $\rho > 1$ , системата е нестабилна и дължината на опашката ще расте неограничено във времето.

### 2.2. Втора система (Компютър 2)

Входящият поток се характеризира с нормално разпределение на интервалите между заявките с  $\mu_A = 3$  s и отклонение, което покрива диапазона  $\pm 1$  s (приемаме  $\sigma \approx 1/3$ , за да може  $3\sigma \approx 1$ ). Времето за обслужване е същото като при първия компютър: равномерно в  $[1, 3]$  s.

Тук  $E[S] = 2$  s, а  $E[A] = 3$  s. Коефициентът на натоварване е  $\rho = \frac{2}{3} \approx 0.67$ . Тъй като  $\rho < 1$ , системата е стабилна.

## III. ИМПЛЕМЕНТАЦИЯ

Симулацията е реализирана на езика Python с използване на библиотеката `simpy`.

### 3.1. Основни параметри

```
1 SIMULATION_TIME = 1800 # 30 minutes
2
3 # Computer 1
4 ARRIVAL_MEAN_1 = 1.0
5 PROCESS_MIN_1 = 1.0
6 PROCESS_MAX_1 = 3.0
7
8 # Computer 2
9 ARRIVAL_MEAN_2 = 3.0
10 PROCESS_MIN_2 = 1.0
11 PROCESS_MAX_2 = 3.0
```

Listing 1: Конфигурация на симулацията

### 3.2. Генериране на заявки

Използват се два генератора за двата компютъра:

```
1 def source_1(env, server):
2     """Source for Computer 1: Exponential Inter-arrival"""
3     while True:
4         yield env.timeout(random.expovariate(1.0 / ARRIVAL_MEAN_1))
5         env.process(process_request(env, server, ...))
6
7 def source_2(env, server):
8     """Source for Computer 2: Normal Inter-arrival"""
9     while True:
10        dt = random.gauss(ARRIVAL_MEAN_2, 0.33) # Sigma approx 1/3
11        if dt < 0: dt = 0
12        yield env.timeout(dt)
13        env.process(process_request(env, server, ...))
```

Listing 2: Генератори на заявки

### 3.3. Реализация на GPSS

Алтернативна реализация на модела чрез езика за симулации GPSS.

```
1 SIMULATE
2 GENERATE 1
3 SEIZE 1
4 ADVANCE 2,1
5 RELEASE 1
6 TERMINATE 0
7 GENERATE 3,1
8 SEIZE 2
9 ADVANCE 2,1
10 RELEASE 2
11 TERMINATE 0
12 GENERATE 1800
13 TERMINATE 1
14 START 1
```

Listing 3: GPSS код на модела

### 3.4. Резултати от GPSS симулация

```
1 GPSS World Simulation Report - Untitled Model 1.10.1
2
3
4 Saturday, February 14, 2026 22:30:26
5
6 START TIME          END TIME  BLOCKS  FACILITIES  STORAGES
7      0.000          1800.000    12        2          0
8
9
10 LABEL              LOC  BLOCK TYPE    ENTRY COUNT  CURRENT COUNT  RETRY
11      1      GENERATE      1800          890        0
12      2      SEIZE         910           0        0
13      3      ADVANCE       910           1        0
14      4      RELEASE       909           0        0
15      5      TERMINATE     909           0        0
16      6      GENERATE      595           0        0
17      7      SEIZE         595           0        0
18      8      ADVANCE       595           1        0
19      9      RELEASE       594           0        0
20     10      TERMINATE     594           0        0
21     11      GENERATE        1           0        0
22     12      TERMINATE        1           0        0
23
24
```

25	FACILITY	ENTRIES	UTIL.	AVE.	TIME	AVAIL.	OWNER	PEND	INTER	RETRY	DELAY
26	1	910	0.999	1.977	1	1217	0	0	0	890	
27	2	595	0.662	2.002	1	2391	0	0	0	0	
28											
29											
30	FEC	XN	PRI	BDT	ASSEM	CURRENT	NEXT	PARAMETER	VALUE		
31	2391	0		1800.170	2391	8	9				
32	1217	0		1800.523	1217	3	4				
33	2394	0		1800.785	2394	0	6				
34	2398	0		1801.000	2398	0	1				
35	2399	0		3600.000	2399	0	11				

## IV. РЕЗУЛТАТИ ОТ СИМУЛАЦИЯТА

Проведена е симулация с продължителност 1800 секунди.

### 4.1. Статистически данни

Таблица 1. Резултати от симулацията

Параметър	Компютър 1	Компютър 2
Обработени заявки	895	597
Средно време на изчакване (s)	444.78	0.016
Максимално време на изчакване (s)	932.10	0.91
Натовареност	99.93%	67.27%

Както се очакваше теоретично, Компютър 1 е претоварен (почти 100% утилизация и огромни времена на чакане), докато Компютър 2 работи в стабилен режим с минимално изчакване.

### 4.2. Сравнение между Python и GPSS

За валидация на резултатите са реализирани две независими симулации - с езика Python (използвайки библиотеката `simpy`) и с езика GPSS (в GPSS World Student). Резултатите от двете симулации са сравнени в Таблица 2.

**Таблица 2. Сравнение на резултатите между Python и GPSS симулациите**

Параметър	Компютър 1		Компютър 2	
	Python	GPSS	Python	GPSS
Обработени заявки	895	910	597	595
Натовареност	99.93%	99.9%	67.27%	66.2%

Както се вижда от таблицата, резултатите от двете симулации са много близки. Малките разлики се дължат на:

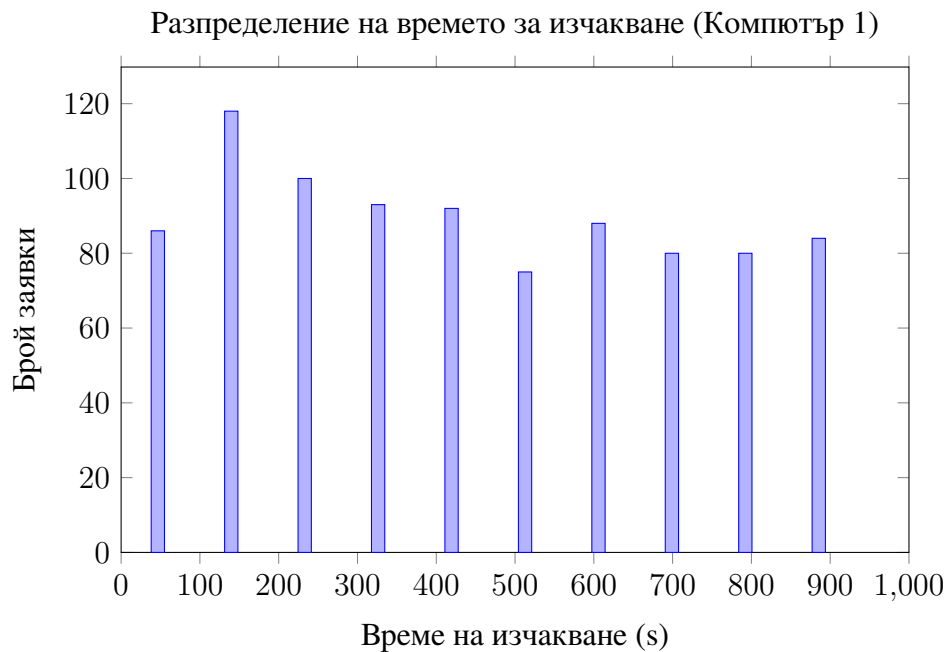
- Случайния характер на входните потоци (различни seed-ове)
- Леко различните реализации на експоненциалното разпределение
- Python симулацията използва пълна реализация на нормалното разпределение, докато GPSS симулацията използва равномерно разпределение за втория компютър ( $3 \pm 1$  s) поради ограниченията на студентската версия

Въпреки тези разлики, двете симулации потвърждават еднакво основните изводи:

1. Компютър 1 е в нестабилен режим ( $\rho > 1$ ) с неограничено растяща опашка
2. Компютър 2 работи стабилно ( $\rho < 1$ ) с минимални времена на чакане

### **4.3. Графична визуализация**

На фигурата по-долу е представена хистограма на времената за изчакване за Компютър 1. Вижда се, че времената за изчакване са разпределени в широк диапазон, което е характерно за нестабилна система.



**Фигура 1. Хистограма на времената за изчакване при Компютър 1**

За Компютър 2 времената за изчакване са пренебрежимо малки (под 1 секунда).

## **V. ЗАКЛЮЧЕНИЕ**

Симулационният модел потвърждава, че при зададените параметри първият компютър не успява да обслужи постъпващия поток от заявки, което води до натрупване на опашка. Вторият компютър се справя успешно със своето натоварване. За оптимизация на системата е препоръчително да се пренасочи част от товара на Компютър 1 към Компютър 2 или да се увеличи производителността на Компютър 1.