Компьютерные сети и телекоммуникации.

Лабораторная работа №1 «Моделирование сервера» (варианты 209/2)

01.04.02 Прикладная математика и информатика
Магистерская программа
«**Технологии программирования и разработки информационно-коммуникационных систем**»

alexander_mikov@mail.ru

Методические указания по выполнению лабораторной работы

1. Требования к проведению исследования.
2. Метод имитационного моделирования.
3. Математическая и программная модели функционирования сервера.

Задание 1. Обработка потока заданий простым сервером

• На сервер поступает поток заданий *InStream*. Задания поступают в моменты времени t_i (i = 1, 2, 3, …); в момент t_0 задания нет. Интервал времени между поступлениями заданий с номерами i – 1 и i равен τ_i . Задание i требует времени σ_i для его обработки сервером *S*.

• Задания, приходящие в систему в то время, когда сервер полностью занят обработкой предыдущих заданий, помещаются в буфер *Q*, характеризующийся предельной длиной (при полной занятости буфера вновь пришедшие задания теряются). Выборка из буфера на обработку производится по одной из дисциплин (FIFO, LIFO, RAND, priority, RoundRobin). Если у сервера несколько ядер, то они работают параллельно и независимо, выбирая задания из общей очереди сразу же, как только заканчивают работу с предыдущим заданием.

• Результаты обработанных заданий образуют выходящий из системы поток *OutStream* (интервалы времени между выходящими результатами обозначим δ_i).

• Величины τ_i и σ_i считаются случайными, взаимно независимыми с функциями распределения вероятностей $A(x) = P\{\tau_i \le x\}$ и $B(x) = P\{\sigma_i \le x\}$ соответственно.

Задание 1. Обработка потока заданий простым сервером

Величины τ_i и σ_i считаются случайными, взаимно независимыми с функциями распределения вероятностей $A(x) = P\{\tau_i \le x\}$ и $B(x) = P\{\sigma_i \le x\}$ соответственно.

Обозначения распределений вероятностей в таблице:

- M – экспоненциальное (пуассоновский поток),
- R[a, b] – равномерное на отрезке [a, b],
- D – детерминированное (постоянное значение), не случайное,
- E_k – Эрланга k-го порядка,
- |Gauss| – распределение $|\gamma|$, где γ имеет нормальное распределение,
- $Gauss^+$ – распределение случайной величины $\max(0, \gamma)$,
- |Gauss|+M – распределение суммы двух случайных величин,
- Rayleigh – распределение Рэлея
- LogNorm – логнормальное распределение (ср. кв. отклонение < 3/2).

	Поток		Сервер S		Буфер Q		Примечание
	A(x)	B(x)	ядер	оконч.	размер	дисциплина	
1	R[1,2]+R[3,4]	Rayleigh	1	T	$\lfloor 1/(1-\rho) \rfloor$	RoundRobin	Δ=1.0
2	E_3	Gauss+	1	K	∞	RAND	$\mathbf{M}\sigma = 2.0$
3	Rayleigh	D = 0.5	$\lceil 1.5/(1-\rho) \rceil$	T	1	-	$\rho < 0.9$
4	M	R[a, b]+1.2	2	K	18	RAND	a>0
5	Gauss+	Rayleigh	4	T	17	LIFO	$\rho = \mathbf{M}\sigma_i/\mathbf{M}\tau_i < 1$
6	E_5+D	|Gauss|	1	K	8	RoundRobin	$\mathbf{M}\sigma = 1.5$
7	R[a, b]	LogNorm	2	T	11	Priority, max σ	b = 2a
8	LogNorm	E_2	1	K	∞	FIFO	$\mathbf{M}\tau = 5.0$

	Поток		Сервер S		Буфер Q		Примечание
	A(x)	B(x)	ядер	окончан.	размер	дисциплина	
9	Rayleigh	|Gauss|	4	K	∞	FIFO	$\mathbf{M}\tau = 4.0$
10	E_3	Gauss+	3	T	3	Priority, max σ	$\mathbf{M}\sigma = 2.0$
11	LogNorm	R[a, 2.0]	2	K	∞	RoundRobin	$\Delta = 0.5$
12	M + D	Rayleigh	2	T	16	RAND	$\mathbf{M}\tau = 10.0$
13	Gauss+	M + Rayleigh	4	K	∞	LIFO	$\rho = \mathbf{M}\sigma_i/\mathbf{M}\tau_i < 1$
14	|Gauss|+D	LogNorm	3	T	6	RoundRobin	$\Delta = \mathbf{M}\sigma/5$
15	R[1.5, b]	|Gauss|	1	K	11	Priority, max σ	$b > 3.0$

Требуется:

- Разработать программу – событийно-ориентированную имитационную модель описанных процессов, провести исследование характеристик компьютерной системы методом имитационного моделирования. Программа должна иметь глобальную переменную «системные часы» (не имеет никакого отношения к часам в компьютере) и счетчик числа заданий для определения момента окончания процесса имитации.

- Имитационная модель реализуется в виде программы на любом универсальном языке программирования (C, C++, C#, Java, Fortran, Python и проч.).

- Исходными данными являются параметры, входящие в формулы для распределений $A(x)$ и $B(x)$, а также время (системное) T, в течение которого работает сервер, или K – количество заданий потока, обработанных сервером.

- При вводе исходных данных (параметров распределений) должен быть обеспечен контроль стационарного режима ($\rho = \mathbf{M}\sigma_i / \mathbf{M}\tau_i < 1$) функционирования моделируемой компьютерной системы для случаев неограниченного буфера и других ограничений.

Требуется:

- Программа, описывающая модель, должна быть дополнена программными конструкциями для проведения вычислительных экспериментов с моделью с использованием генераторов псевдослучайных чисел, а также для обработки результатов вычислительных экспериментов и получения результатов исследования модели.
- В таблице ниже указано, что должно быть результатами исследования.
- Для получения результатов исследования с достаточной точностью, как правило, необходимо провести не менее 10000 вычислительных экспериментов.

Результаты моделирования

	Исследуемые характеристики		Исследуемые характеристики
1	$W(x)$, $P_{busy}(x)$	9	$P_{idle}(x)$, $A_{out}(x)$
2	$P_{idle}(x)$, $P_{отказа}$	10	$W(x)$, $P_{отказа}$
3	$Core(n)$, $P_{отказа}$	11	$U(x)$, $P_{простоя}$
4	$Core(n)$, $W(x)$	12	$P_{busy}(x)$, $D\omega_i$
5	$L(n)$, $P_{простоя}$	13	$P_{idle}(x)$, $Corr(\delta_i , \omega_i)$
6	$U(x)$, $P_{простоя}$	14	$U(x)$, $P_{idle}(x)$
7	$L(n)$, $A_{out}(x)$	15	$P_{busy}(x)$, $L(n)$
8	$W(x)$, $Corr(\delta_i , \omega_i)$		

Здесь:

- $W(x) = P\{\omega_i \le x\}$ – функция распределения вероятностей времени ожидания ω_i в очереди произвольного задания,
- $U(x)$ – функция распределения вероятностей времени пребывания произвольного задания в системе (от поступления до окончания обработки),
- $L(n)$ – распределение вероятностей длины очереди, т.е. $L(n) = P\{\text{длина очереди} = n\}$,
- $P_{\text{отказа}}$ – вероятность отказа заданию в обработке,
- $P_{\text{простоя}}$ – вероятность простоя сервера в произвольный момент времени t,
- $P_{busy}(x)$ – функция распределения вероятностей времени непрерывной занятости сервера (одним или несколькими последовательными заданиями),
- $Core(n)$ – распределение вероятностей количества занятых ядер,
- $P_{idle}(x)$ – функция распределения вероятностей времени непрерывного простоя сервера,
- $A_{out}(x) = P\{\delta_i \le x\}$ – функция распределения вероятностей длительностей интервалов между последовательными моментами выхода заданий из системы (с сервера),
- Corr – коэффициент корреляции двух связанных случайных величин,
- $D\delta_i$ – дисперсия δ_i .

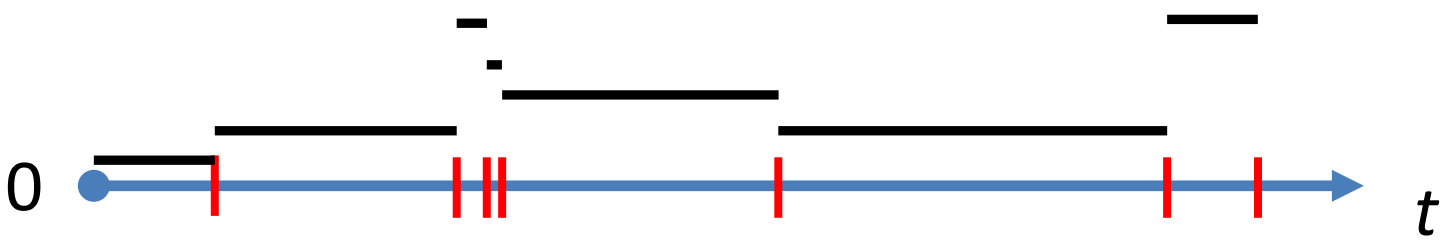
Метод дискретного событийно-ориентированного моделирования

- Это метод анализа процессов в системе, развивающихся во времени.
- Предполагается, что **все** изменения в системе происходят в отдельные моменты времени, отстоящие друг от друга на конечный интервал (дискретность):

- В остальные моменты времени в системе ничего не происходит (заслуживающего внимания с точки зрения цели исследования).

Метод дискретного событийно-ориентированного моделирования

- То, что происходит в «отдельные моменты времени», называется событиями.
- Событие состоит в том, что изменяется **состояние** системы.



- Считается, что изменение происходит мгновенно (хотя это абстракция – противоречит законам физики).

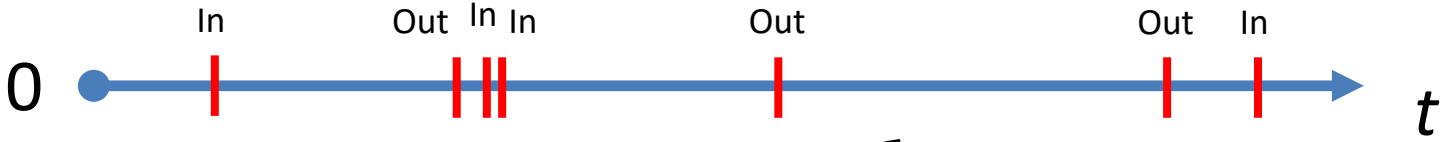
Метод дискретного событийно-ориентированного моделирования

- Событие – это некоторое преобразование состояния в данный момент времени, т.е. функция (с точки зрения программирования).
- Хотя событий происходит очень много (может быть, неограниченно много), но некоторые события однотипны, поэтому они могут быть заданы одной и той же функцией (но с разными фактическими параметрами).
- Важно: Во всех этих преобразованиях участвует системное время (t), представленное «системными часами».

Метод дискретного событийно-ориентированного моделирования

Как «идут» системные часы?

- У каждого события есть момент времени, когда оно «совершается». События можно упорядочить по времени свершения (здесь In, Out – имена типов событий).



- «Стрелка» системных часов будет «прыгать» по этим упорядоченным моментам времени от 0 до заданного момента окончания моделирования.

Метод дискретного событийно-ориентированного моделирования

Как определить моменты совершения событий?

- Это уже зависит от конкретной системы, которую мы моделируем, от выделенных типов событий, от причинно-следственных связей между событиями.
- Например, событие Out не может произойти, пока не произойдет событие In.

- Если у нас есть поток входных заданий для сервера, то в момент поступления i-го задания t_i можно определить момент поступления (i +1)-го задания, прибавив к t_i величину τ.

Метод дискретного событийно-ориентированного моделирования

- События обрабатываются (выполняются функции, связанные с событиями) в соответствии с возрастанием назначенного им системного времени.
- В общем случае при обработке события могут быть запланированы новые события на «будущие» моменты системного времени. Поэтому, имеется пул «будущих» событий.
- Из этого пула выбирается событие с минимальным временем, на это время устанавливаются часы, а событие становится очередным обрабатываемым.

Программное (имитационное) моделирование

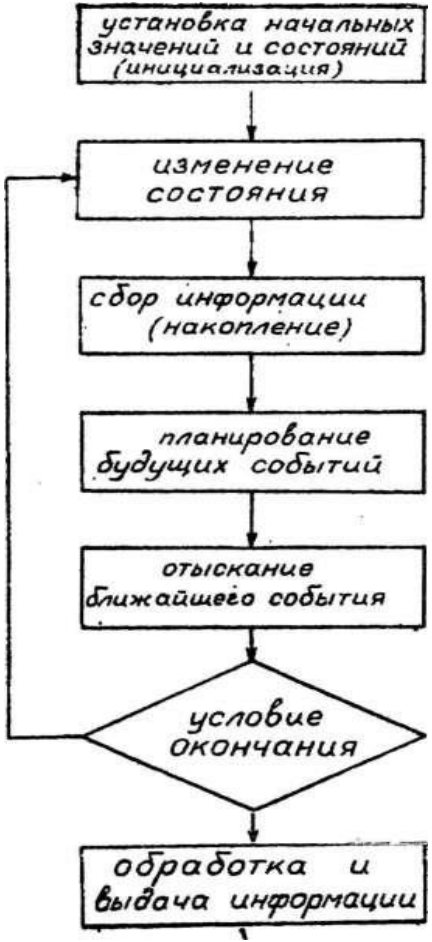
При разработке имитационной модели возникает ряд задач:

1. Отображение моделируемой компьютерной сети на память и разработка алгоритмов моделирующей ЦВМ. Введение необходимых структур данных и обрабатывающих их функций.

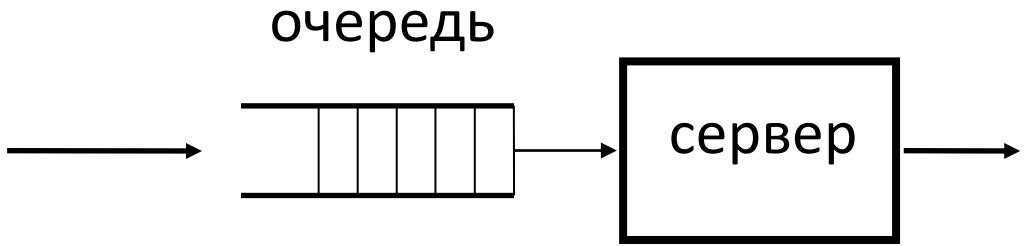
2. Представление времени в модели.

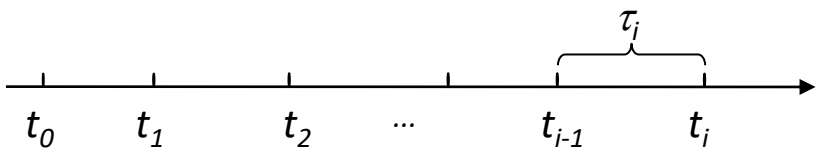
3. Выбор множества комбинаций управляемых данных (соответствующих различным условиям работы) для получения максимальной информации об исследуемой сети при минимальных затратах на моделирование. Выбранные комбинации должны позволить нам не упустить важных аспектов поведения сети, проявляющихся не при всех комбинациях.

4. Обработка и интерпретация результатов (заключение о свойствах сети на основе ограниченного количества экспериментов).

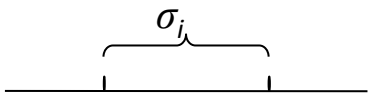

установка начальных значений и состояний (инициализация)
изменение состояния
сбор информации (накопление)
планирование будущих событий
отыскание ближайшего события
условие окончания
обработка и выдача информации

Модель сервера в сети как системы массового обслуживания

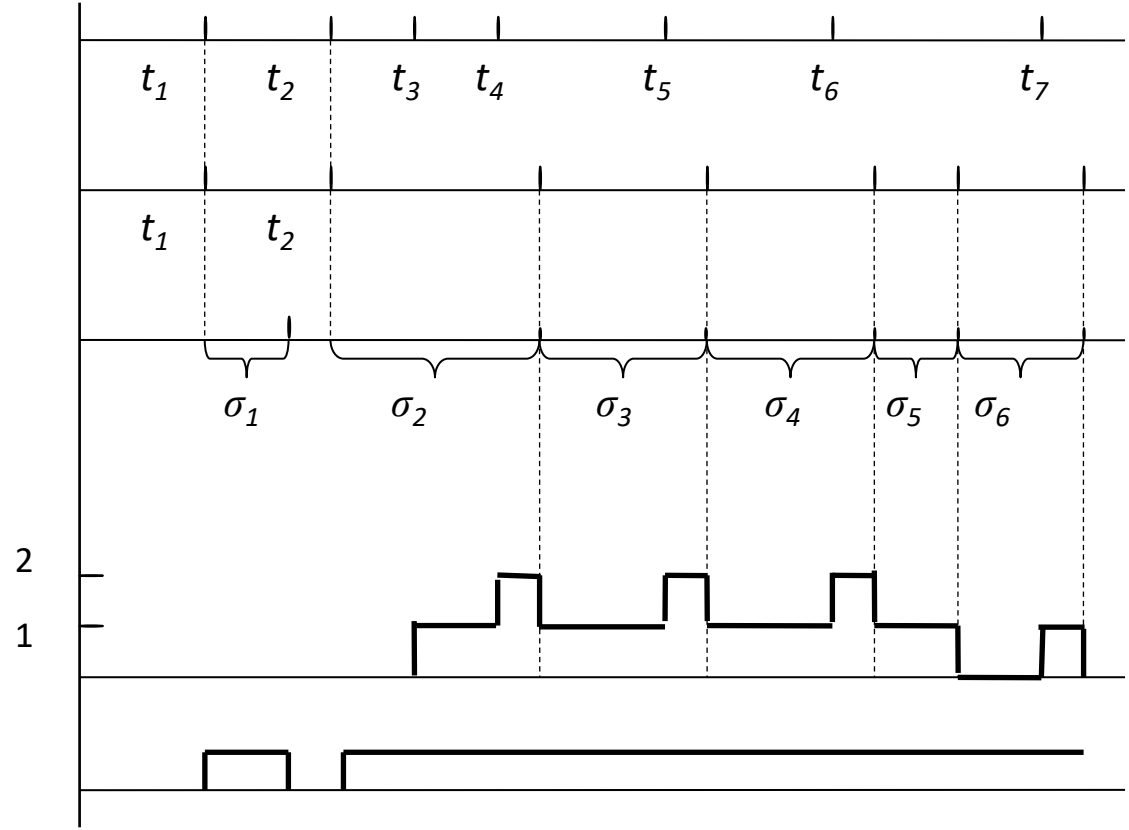




входной поток



время обработки (обслуживания)



Случайный поток: поступление заданий в систему

Моменты времени поступления заданий на сервер

Моменты времени окончания обработки заданий

Случайный процесс «длина очереди» (изменение состояния)

Случайный процесс «занятость сервера» (изменение состояния)

Имитационное моделирование

- Построим систему массового обслуживания внутри машины.
- Пусть мы имеем программу, обращаясь к которой получаем реализацию случайных величин τ_i и σ_i с заданными распределениями вероятностей $A(x)$ и $B(x)$.

Рассмотрим решение задачи отыскания длины очереди.

Состояние СМО опишем парой целых чисел (K, L). Первое число относится к каналу (серверу),

$$K = \begin{cases} 0, & \text{канал свободен} \\ 1, & \text{канал занят} \end{cases}$$

второе (L) – длина очереди, $L \geq 0$.

Состояние меняется не непрерывно, а только в некоторые дискретные моменты времени. Множество этих моментов можно разбить на классы однотипных событий. Присвоим классам следующие идентификаторы:

1) T1 – моменты времени прихода заданий в СМО;

2) T2 – моменты окончания обработки в сервере.

Переменная TS будет обозначать системное время. Состояния изменяются различным образом в моменты типа T1 и T2.

- T1. Пришло новое задание. Оно может занять сервер, если он был перед этим свободен (если $K == 0$, то $K = 1$), или встать в очередь, если сервер занят (если $K == 1$, то $L = L + 1$).
- T2. В момент окончания обработки сервер освобождается, если очередь пуста (если $L == 0$, то $K = 0$), или очередь уменьшается на единицу за счет задания, переходящего в сервер (если $L \neq 0$, то $L = L - 1$).

- Теперь мы знаем, какие преобразования пары (*K*, *L*) необходимы в моменты T1 и T2. Остается определить последовательность их чередования.
- Положим, в нулевой момент времени в СМО нет заданий. Тогда, очевидно, следующий момент изменения состояния будет T1.
- Пусть есть некая программа, позволяющая получать (генерировать) реализации случайной величины τ – интервалов времени между приходами заданий.
- Тогда мы сможем генерировать первый интервал τ_1 и присвоить это значение переменной T1. Переменной T2 мы присвоим значение T1 + σ_1 , где σ_1 – сгенерированная продолжительность обработки первого задания.
- Кроме того, TS := T1. Системное время указывает, в какой момент мы рассматриваем систему и делит значения всех остальных временных переменных на настоящие и будущие.

- На первом шаге алгоритма имитации T1 является настоящим моментом, а T2 – будущим.
- Обработав момент T1, т. е. изменив состояние (*K*, *L*) мы должны спланировать будущий момент прихода следующего задания: T1: = T1 + τ.
- Теперь (по отношению к текущему системному времени TS) у нас два будущих момента: T1 и T2. Какой из них обрабатывать раньше? Очевидно, тот, который раньше наступит. Передвигаем стрелку системных часов, TS := min(T1, T2).

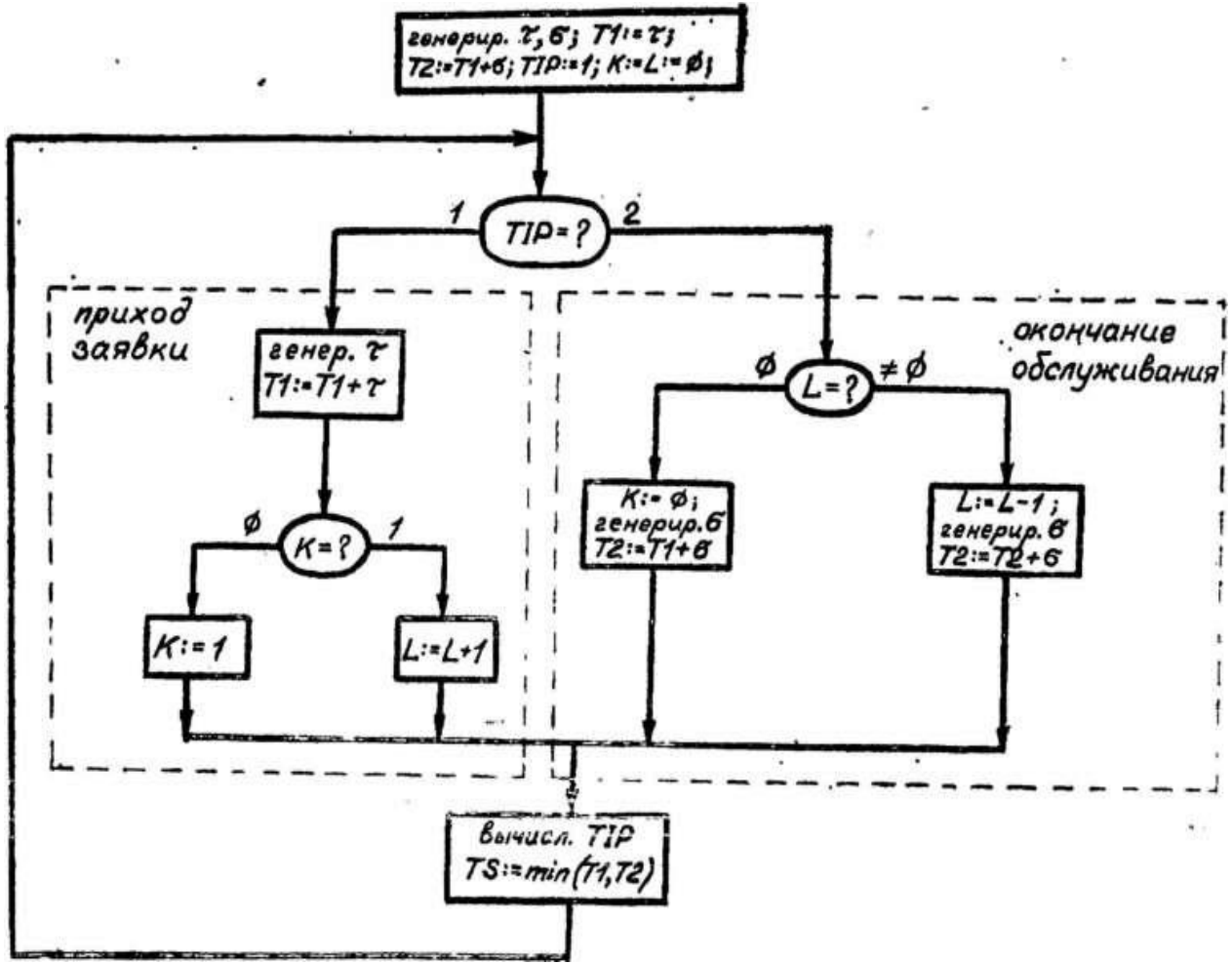
Для удобства можно ввести еще рабочую переменную

TIP = 1, если T1 $\leq$ T2; TIP = 2, если T1 > T2,

указывающую тип следующего момента времени.

Обрабатывая момент окончания обслуживания, мы должны спланировать новый момент T2. Если очередь не пуста, то новое задание сразу поступит в сервер и T2: = T2 + σ_1. Если в очереди нет заданий, то необходимо дождаться их прихода, т. е.

T2: = T1 + σ_1.


генерир. τ, б; T1:=τ;
T2:=T1+б; TIP:=1; K:=L:=ϕ;
TIP=?
1
2
приход заявки
генер. τ
T1:=T1+τ
K=?
ϕ
1
K:=1
L:=L+1
окончание обслуживания
L=?
ϕ
≠ϕ
K:=ϕ;
генерир.б
T2:=T1+б
L:=L-1;
генерир.б
T2:=T2+б
вычисл. TIP
TS:=min(T1,T2)