1. Общие сведения

Целью курсовой работы является получение навыков моделирования систем массового обслуживания (СМО). Развитие человеческих взаимоотношений привело к тому, что практически ежедневно сталкиваемся с СМО. Это неизбежно, т. к. используя результаты труда других людей, желаем это делать в удобное для себя время. В результате возникает все больше систем со случайными потоками требований и дисциплинами обслуживания.

1.1. Варианты заданий

Вариант задания выбирается по последним двум цифрам шифра студенческого билета, если последние две цифры больше количества вариантов, то номер варианта выбирается по сумме двух последних цифр шифра.

Вариант 1. На сборочный участок цеха предприятия через интервалы времени, распределенные экспоненциально со средним значением 18 мин., поступают партии, каждая из которых состоит из трех одинаковых деталей. Треть всех поступающих деталей перед сборкой должна пройти предварительную обработку в течение 7 мин. На сборку подаются только обработанные детали. Длительность сборки – 4 мин. Затем изделия (собранные из трех деталей) поступают на регулировку, продолжающуюся в среднем 8 мин (время выполнения этой операции распределено экспоненциально). В процессе регулировки возможно выявление 5% бракованных изделий, которые разбираются на составные детали и направляются снова на предварительную обработку.

Смоделировать работу участка в течение 24 часов. Определить возможные места появления очередей и их вероятностно-временные характеристики, выявить причины их возникновения, предложить меры их устранения и смоделировать скорректированную систему.

Вариант 2. В студенческой вычислительной лаборатории расположены три компьютера. На двух из них установлены пакеты математических вычислительных программ, а на третьем офис, а еще к третьему компьютеру подключен принтер. Студенты приходят с интервалом в 30±2 мин, и одна треть из них хочет, и рассчитать свои задания, и распечатать результаты, а остальные – только рассчитать задания. Допустимая очередь в вычислительной лаборатории составляет 3 человека. Вычисления (работа на одном из двух первых компьютеров) занимают 15±10 мин, а оформление и распечатка (работа на третьем компьютере) –30±15 мин. Кроме того, 10% студентов, распечатавших результаты, сразу же находят у себя ошибки и возвращаются для повторного выполнения задания. Однако если лаборатория переполнена, им приходится уйти.

Смоделировать работу вычислительной лаборатории в течение 60 часов. Определить загрузку компьютеров, вероятность отказа в обслуживании вследствие переполнения очереди. Определить соотношение в очереди желающих вычислять и распечатывать результаты.

Вариант 3. В системе передачи данных осуществляется обмен пакетами данных между пунктами А и В по дуплексному каналу связи. Пакеты поступают в пункты системы от абонентов с интервалами времени между ними 10±3 мс. Передача пакета занимает 10 мс. В пунктах имеются буферные регистры, которые могут хранить два пакета (включая передаваемый). В случае прихода пакета в момент занятости регистров, пунктам системы предоставляется выход на спутниковую полудуплексную линию свя-

зи, которая осуществляет передачу пакетов данных за 10±5 мс. При занятости спутниковой линии пакет получает отказ.

Смоделировать обмен информацией в системе передачи данных в течение 1 мин. Определить частоту вызовов спутникового канала и его загрузку. В случае возможности отказов определить необходимый для безотказной работы системы объем буферных регистров.

Вариант 4. Вычислительная система включает три ЭВМ. В систему, в среднем, через 30 с поступают задания, которые попадают в очередь на обработку к первой Распределенный банк данных системы сбора информации организован на базе ЭВМ, соединенных дуплексным каналом связи. Поступающий запрос обрабатывается на первой ЭВМ и с вероятностью 50% необходимая информация обнаруживается на месте. В противном случае необходима посылка запроса во вторую ЭВМ. Запросы поступают на первую ЭВМ через 20±3 с, первичная обработка запроса занимает 1 с, выдача ответа требует 20±2 с, передача по каналу связи занимает 2 с. Временные характеристики второй ЭВМ аналогичны первой.

Смоделировать прохождение 400 запросов. Определить необходимую емкость накопителей перед ЭВМ, обеспечивающую безотказную работу системы, и функцию распределения времени обслуживания заявки.

Вариант 5. В некоторое цифровое устройство через 3±1 мин. поступают задания длиной 600±200 Кбайт. Скорость ввода, вывода и обработки заданий – 100 Кбайт/сек. Задания проходят последовательно ввод, обработку и вывод, буферируясь перед каждой операцией. После вывода 5% заданий оказываются выполненными неправильно вследствие сбоев и возвращаются на ввод. Для ускорения обработки задания в очередях располагаются по возрастанию их длины, т. е. короткие задания обслуживают в первую очередь.

Смоделировать работу устройства в течение 3 ч. Определить необходимую емкость буферов и функцию распределения времени обслуживания заданий (построить гистограмму).

Вариант 6. Вычислительная система включает три ЭВМ. В систему, в среднем, через 50 с поступают задания, которые попадают в очередь на обработку к первой ЭВМ, где они обрабатываются около 50 с (и там и там экспоненциальное распределение). После этого задание поступает одновременно во вторую и третью ЭВМ. Вторая ЭВМ может обработать задание за 14±5 с, а третья – за 16±1 с (равномерное распределение). Окончание обработки задания на любой ЭВМ означает снятие ее с решения с той и другой машины.

Смоделировать 4 ч работы системы. Определить необходимую емкость накопителей перед всеми ЭВМ, коэффициенты загрузки ЭВМ и функцию распределения времени обслуживания заданий (построить гистограмму).

Вариант 7. К ЭВМ подключено четыре терминала, с которых осуществляется решение задач. По команде с терминала выполняют операции редактирования, трансляции, планирования и решения. Причем, если хоть один терминал выполняет планирование, остальные вынуждены простаивать из-за нехватки оперативной памяти. Если два терминала выдают требование на решение, то оставшиеся два простаивают, и если работают три терминала, выдающих задания на трансляцию, то оставшийся терминал блокируется. Интенсивности поступления задач различных типов равны. Задачи одного

типа от одного терминала поступают через экспоненциально распределенные интервалы времени со средним значением 300 с. Выполнение любой операции длится 20 с.

Смоделировать работу ЭВМ в течение 8 ч. Определить загрузку процессора, вероятности простоя терминалов и частоту одновременного выполнения трансляции с трех терминалов.

Вариант 8. Детали, необходимые для работы цеха, находятся на цеховом и центральном складах. На цеховом складе хранится 30 комплектов деталей, потребность в которых возникает через 60±10 мин и составляет один комплект. В случае снижения запасов до трех комплектов формируется в течение 60 мин заявка на пополнение запасов цехового склада до полного объема в 30 комплектов, которая посылается на центральный склад, где в течение 60±20 мин происходит комплектование и за 60±5 мин осуществляется доставка деталей в цех.

Смоделировать работу цеха в течение 600 ч. Определить вероятность простоя цеха из-за отсутствия деталей и среднюю загрузку цехового склада. Определить момент пополнения запаса цехового склада, при котором вероятность простоя цеха будет равна 0.

Вариант 9. В студенческой вычислительной лаборатории расположены компьютеры: две рабочие станции и сервер. Студенты приходят с интервалом в 15±5 мин, и одна треть из них хочет работать и на том, и на другом, а остальные – используют только рабочие станции. Допустимая очередь в вычислительной лаборатории составляет 5 человек. Работа на сервере занимает 10±5 мин, а на рабочей станции – 30±10 мин. Кроме того, 15% работающих на компьютерах возвращаются для повторной работы (в случае, если очередь переполнена, они тоже получают отказ).

Смоделировать работу вычислительной лаборатории в течение 60 часов. Определить загрузку сервера и рабочих станций, вероятность отказа в обслуживании вследствие переполнения очереди. Определить соотношение в очереди желающих работать на сервере и на рабочих станциях.

Вариант 10. Для обеспечения надежности АСУ ТП в ней используется две ЭВМ. Первая ЭВМ выполняет обработку данных о технологическом процессе и выработку управляющих сигналов, а вторая находится в "горячем резерве". Данные в ЭВМ поступают через 10±5 с, обрабатываются в течение 5 с, затем посылается управляющий сигнал, поддерживающий заданный темп процесса. Если к моменту посылки следующего набора данных не получен управляющий сигнал, то интенсивность выполнения технологического процесса уменьшается вдвое и данные посылаются через 20±3 с. Основная ЭВМ каждые 30 с посылает резервной ЭВМ сигнал о работоспособности. Отсутствие сигнала означает необходимость включения резервной ЭВМ вместо основной. Характеристики обеих ЭВМ одинаковы. Подключение резервной ЭВМ занимает 12 с, после чего она заменяет основную до восстановления, а процесс возвращается к нормальному темпу. Отказы ЭВМ происходят через 300±30 с. Восстановление занимает 200 с. Резервная ЭВМ абсолютно надежна.

Смоделировать 1 ч работы системы. Определить среднее время нахождения технологического процесса в заторможенном состоянии и среднее число пропущенных из-за отказов данных.

Вариант 11. На вычислительном центре в обработку принимают три класса заданий A, B, C. Исходя из наличия оперативной памяти ЭВМ задания классов A и B могут решаться одновременно, а задания класса C монополизируют ЭВМ. Задания класса A поступают в систему через 30±20 мин, класса B – через 30±10 мин, класса C – 60±30

мин. И требуют для выполнения: класса А – 25±5 мин, класса В – 32±3 мин и класса С – 28±5 мин. Задачи класса С загружаются в ЭВМ, если она полностью свободна. Задачи классов А и В могут дозагружаться к решающейся задаче.

Смоделировать работу ЭВМ за 80 часов. Определить ее загрузку.

Вариант 12. В системе передачи цифровой информации передается речь в цифровом виде. Речевые пакеты передаются через два транзитных канала, буферируясь в накопителях перед каждым каналом. Время передачи пакета по каналу составляет 2 мс. Пакеты поступают через 3±1 мс. Пакеты, передававшиеся более 10 мс, на выходе системы уничтожаются, так как их появление в декодере значительно снизит качество передаваемой речи. Уничтожение более 25% пакетов недопустимо. При достижении такого уровня система за счет подключения ресурсов ускоряет передачу до 4 мс на канал. При снижении уровня до приемлемого происходит отключение ресурсов.

Смоделировать 50 с работы системы. Определить частоту уничтожения пакетов и частоту подключения ресурса.

Вариант 13. В компьютерный класс с интервалом времени 15±5 мин заходят студенты, желающие обработать на компьютере результаты лабораторной работы. В классе для этого предназначен всего один компьютер. Время, необходимое для решения задач, включая вывод результатов на печать, характеризуется интервалом 15±10 мин. Третья часть пользователей после окончания решения своей задачи производит запись своей программы на внешний магнитный носитель (продолжительность этой операции – 3±2 мин). В классе не допускается присутствие более семи человек.

Смоделировать процесс обслуживания 300 пользователей. Подсчитать число пользователей, не нашедших свободного места в очереди. Определить среднее число пользователей в очереди, а также коэффициент загрузки компьютера.

Вариант 14. Из литейного цеха на участок обработки и сборки поступают заготовки через 20±5 мин. Треть из них обрабатывается в течение 30 мин (получаются детали первого типа) и поступает на комплектацию. Две трети заготовок обрабатывается за 20 мин (получаются детали второго типа) перед комплектацией, которая требует наличия одной детали первого типа и двух деталей второго. После этого все три детали подаются на сборку, которая занимает 40±2 мин для первой детали и 35±8 мин для двух других, причем они участвуют в сборке одновременно. При наличии на выходе одновременно всех трех деталей изделие покидает участок.

Смоделировать работу участка в течение 100 ч. Определить места образования и характеристики возможных очередей.

Вариант 15. Система автоматизации проектирования состоит из ЭВМ и трех терминалов. Каждый проектировщик формирует задание на расчет в интерактивном режиме. Набор строки задания занимает 12±5 с. Получение ответа на строку требует 2 с работы ЭВМ и 6 с работы терминала. После набора десяти строк задание считается сформированным и поступает на решение, при этом в течение 14±3 с ЭВМ прекращает выработку ответов на вводимые строки. Вывод результата требует 8 с работы терминала. Анализ результата занимает у проектировщика 20 с, после чего цикл повторяется.

Смоделировать работу системы в течение 10 ч. Определить вероятность простоя проектировщика из-за занятости ЭВМ и коэффициент загрузки ЭВМ.

Вариант 16. Магистраль передачи данных состоит из двух каналов основного и резервного и общего накопителя. При нормальном функционировании сообщения пере-

даются по основному каналу за 10±3 с. В основном канале происходят сбои через интервалы времени 240±35 с. Если сбой происходит во время передачи, то за 2 с запускается резервный канал, который передает прерванное сообщение с самого начала. Восстановление основного канала занимает 25±5 с. После восстановления резервный канал выключается и основной канал продолжает работу с очередного сообщения. Сообщения поступают через 9±4 с и остаются в накопителе до окончания передачи. В случае сбоя передаваемое сообщение передается повторно по запасному каналу.

Смоделировать работу магистрали передачи данных в течение 2 часов. Определить загрузку запасного канала, частоту отказов канала и число прерванных сообщений. Определить функцию распределения времени передачи сообщений по магистрали.

Вариант 17. В узел коммутации сообщений, состоящий из входного буфера, процессора, двух исходящих буферов и двух выходных линий, поступают сообщения с двух направлений. Сообщения с одного направления поступают во входной буфер, обрабатываются в процессоре, буферизуются в выходном буфере первой линии и передаются по выходной линии. Сообщения ее второго направления обрабатываются аналогично, но передаются по второй выходной линии. Применяемый метод контроля потоков требует одновременного присутствия в системе не более трех сообщений на каждом направлении. Сообщения поступают через интервалы 20±7 мс. Время обработки в процессоре равно 10 мс на сообщение, время передачи по выходной линии равно 25±5 мс. Если сообщение поступает при наличии трех сообщений в направлении, то оно получает отказ.

Смоделировать работу узла коммутации в течение 100 с. Определить загрузки устройств и вероятность отказа в обслуживании из-за переполнения буфера направления. Определить изменения в функции распределения времени передачи при снятии ограничений, вносимых методом контроля потоков.

Вариант 18. На комплексный конвейер сборочного цеха каждые 15±1 мин поступают 10 изделий первого типа и каждые 20±7 мин поступают 15 изделий второго типа. Конвейер состоит из секций, вмещающих по 10 изделий каждого типа. Комплектация начинается только при наличии деталей обоих типов в требуемом количестве и длится 10 мин. При нехватке деталей секция конвейера остается пустой в течение 10 минут.

Смоделировать работу конвейера сборочного цеха в течение 8 часов. Определить вероятности пропуска секции, средние, максимальные очереди каждого типа изделий. Определить экономическую целесообразность перехода на секции по 20 изделий с временем комплектации 20 мин.

Вариант 19. Специализированная вычислительная система (ВС) состоит из 3-х процессоров и общей оперативной памяти. Задания, поступающие на обработку через интервалы времени 10±2 мин, занимают объем оперативной памяти размером в страницу. После трансляции первым процессором в течении 6±1 мин их объем увеличивается до двух страниц и они поступают в оперативную память. Затем после редактирования во втором процессоре, которое занимает 2±0.5 мин на страницу, объем возрастает до четырех страниц. Отредактированные задания через оперативную память поступают в третий процессор на решение, требующее 2±0.4 мин на страницу, и покидают систему, минуя оперативную память.

Смоделировать работу ВС в течение 50 часов. Определить характеристики занятости оперативной памяти по всем видам операций.

Вариант 20. ЭВМ обслуживает три терминала по круговому циклическому алгоритму, предоставляя каждому терминалу 50 с. Если в течение этого времени задание обрабатывается, то обслуживание завершается; если нет, то остаток задачи становится в специальную очередь, которая использует свободные циклы терминалов, т. е. задача обслуживается, если на каком-либо терминале нет заявок. Заявки на каждый из терминалов поступают через 150±40 с и имеют длину 600+50 знаков. Скорость обработки заданий ЭВМ равна 10 знаков/с.

Смоделировать 10 ч работы ЭВМ. Определить загрузку ЭВМ, параметры очереди неоконченных заданий. Определить минимальную величину цикла терминала, при которой все заявки будут обслужены без специальной очереди.

1.2. Краткие сведения о системах массового обслуживания

В общем случае СМО классифицируется по следующим признакам:

- закону распределения входного потока
- числу обслуживающих приборов
- закону распределения времени обслуживания в обслуживающих приборах
- числу мест в очереди
- дисциплине обслуживания

При определении любой системы обслуживания для сокращенной записи используется следующая система кодировки, в которой на месте букв ставится соответствующая характеристика СМО (табл. 1):

A | B | C | D | E

Табл. 1. Характеристики СМО

A	Закон распределения интервала времени между поступлением заявок	**M** – экспоненциальный **E** – Эрланга **H** – гиперэкспоненциальный **Г** – гамма-распределение **D** – детерминированное распределение **G** – произвольное распределение
B	Закон распределения времени обслуживания в приборах СМО	
C	Число обслуживающих приборов	**1** – для одноканальных систем ***l*** – для многоканальных систем
D	Число мест в очереди. Если число мест не ограничено, то поле можно опустить.	***r*** либо ***n*** – для конечного числа мест
E	Дисциплина обслуживания. По умолчанию **LIFO** – в этом случае поле может опускаться	**FIFO**, **LIFO**, **RANDOM**

Примеры.

1. **M | M | 1** – СМО с одним обслуживающим прибором, бесконечной очередью, экспоненциальным законом распределения интервалов времени между поступлением заявок и временем обслуживания и дисциплиной обслуживания FIFO.

2. **E | H | *l* | *r* | LIFO** – СМО с несколькими обслуживающими приборами, конечной очередью, законом распределения Эрланга интервалов между поступлением заявок, гиперэкспоненциальным законом распределения времени обслуживания заявок в приборах и дисциплиной обслуживания LIFO.

3. **G | G | *l*** – СМО с несколькими приборами, бесконечной очередью, произвольными законами распределения времени между поступлением заявок и времени обслуживания, FIFO.

СМО состоит из одного или более обрабатывающих устройств (сервисов), обслуживающих прибытие сущностей, ещё называемых требованиями или фишками, в систему [1]. Сущности (требования) – это индивидуальные элементы, обрабатываемые в системе. Сущность, находящая сервис занятым, встает в очередь перед сервисом (обрабатывающим устройством). Сущности представляют собой описание динамических процессов в реальных системах. Они могут описывать как реальные физические объекты, так и нефизические объекты. Сущностями могут быть: клиенты, обслуживаемые в ресторане, больнице, аэропорту; документы, части, которые должны быть обслужены или обработаны. В бизнес-процессах – это документы или электронные отчеты (чеки, заказы, контракты). В производственных моделях, сущностями являются сырье, компоненты или готовая продукция. Кроме этого, под сущностями понимают различные типы объектов, типы пакетов данных в сети, данные в программных пакетах. В табл. 2 приведены элементы СМО [2, 3, 4].

Табл. 2. Основные элементы СМО

Название элемента СМО	Назначение элемента СМО
Генераторы	Генерируют поступление сущностей в систему и временные интервалы их прибытия
Обрабатывающие устройства (сервисы)	Количество обрабатывающих устройств в системе, количество очередей, время обработки одной сущности
Очередь	Правило, по которому обрабатывающее устройство выбирает сущность для обслуживания

В зависимости от поведения сущности, поступившей в систему обслуживания в момент, когда все обрабатывающие устройства заняты, СМО делятся на три группы:

- системы с отказами, или системы с потерями;
- системы с ожиданием, или системы без потерь;
- системы смешанного типа.

В системах с отказами (системах с потерями) любая вновь поступившая сущность на обслуживание, застав все сервисы занятыми, покидает систему. Примером системы с отказами может служить работа автоматической телефонной станции: абонент получает отказ, если необходимая линия связи занята.

В системах с ожиданием (системах без потерь) сущность, поступившая в систему, может её покинуть только после того, как будет обслужена. В таких системах сущности, поступившие в момент, когда все сервисы заняты, образуют очередь. Примером системы обслуживания без потерь является система ремонта техники связи: неисправная техника не может быть использована без ремонта.

В системах смешанного типа сущность, поступившая, когда все сервисы заняты, некоторое время ожидает в очереди, и если за это время не принимается к обслуживанию, то покидает систему. Примером такой системы является обслуживание абонента в переговорном зале междугородной телефонной станции (МТС): абоненту разговор должен быть предоставлен в течение 1 часа. Если за это время разговор не состоялся, то, как правило, абонент покидает МТС.

По числу обрабатывающих устройств (сервисов) различают: одноканальные СМО (рис. 1) и многоканальные СМО (рис. 2).

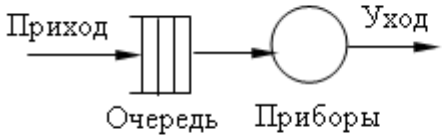


Рис. 1. Одноканальная СМО

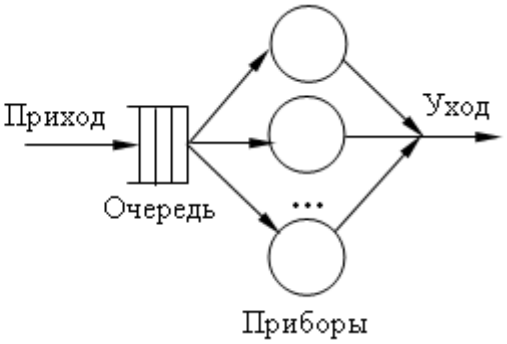


Рис. 2. Могоканальные СМО

В свою очередь, многоканальные системы могут состоять из однотипных и разнотипных (по пропускной способности) каналов.

По числу сущностей, которые могут одновременно находиться в обслуживающей системе, различают системы с ограниченным и неограниченным потоком требований.

Существуют системы обслуживания, в которых обрабатывающие устройства расположены последовательно (пронумерованы). Очередное требование поступает сначала на первое из них и лишь в том случае, если оно занято, передается второму и т. д. Такие системы называются упорядоченными. Все остальные системы обслуживания, в которых требования распределяются между обрабатывающими устройствами по любому другому принципу, относятся к числу неупорядоченных систем.

По характеру источника сущностей (генератора) различают СМО с конечным и бесконечным количеством требований на входе, соответственно различают замкнутые и разомкнутые СМО. В первом случае в системе циркулирует конечное, обычно постоянное количество требований, которые после завершения обслуживания возвращаются в генератор.

Кроме того, все СМО можно разделить по дисциплине обслуживания [27]. Дисциплина обслуживания определяется правилом, которое устройство обслуживания использует для выбора из очереди следующего требования (если таковые есть) по завершении обслуживания текущего требования. Обычно используются такие дисциплины очереди:

- FIFO (First-In, First-Out): требования обслуживаются по принципу «первым прибыл – первым обслужен»;
- LIFO (Last-In, First-Out): требования обслуживаются по принципу «последним прибыл – первым обслужен»;
- приоритет: требования обслуживаются в порядке их значимости.

Современные задачи анализа и оптимизации процессов массового обслуживания сложны и требуют реализации дополнительных атрибутов запросов, условий срабатывания, различного времени обслуживания и т. д., в связи с этим без современных программных средствах и пакетов не обойтись.

1.2.1. Основные понятия СМО

Для рассмотрения системы введем следующие определения:

Очередь - это группа заявок, ожидающих обслуживания.

Модельное время - это промежуток времени между началом моделирования и его завершением.

Простая система массового обслуживания, изображенная на рис. 1, характеризуется двумя независимыми случайными переменными:

интервал прибытия заявок – это интервал времени между последовательными моментами прибытия заявок в систему;
время обслуживания – это время, требуемое прибору для выполнения обслуживания.

Величины, характеризующие работу данной системы обслуживания и зависящие от двух вышеперечисленных независимых случайных переменных, могут стать предметом исследования. Ниже перечислены некоторые из этих случайных величин:

1. Число заявок, прибывших на обслуживание за заданный промежуток времени.
2. Число заявок, которые попали на обслуживание сразу же по прибытии (минуя очередь).
3. Среднее время пребывания заявок в очереди.
4. Средняя длина очереди.
5. Максимальная длина очереди.
6. Нагрузка прибора, являющаяся функцией времени, которое потрачено прибором на обслуживание в течение заданного промежутка времени.

Следует заметить, что разработку логической схемы модели на ЭВМ, которая будет имитировать систему обслуживания с одним прибором и очередью, нужно вести при следующих условиях:

1. Случайные переменные *интервал прибытия* и *время обслуживания* являются равномерно распределенными и принимают только целые значения.
2. Все прибывающие заявки должны быть обслужены независимо от длины очереди.
3. Вначале моделирования система "пуста", т. е. нет очереди и обслуживающий прибор свободен.
4. Моделирование продолжается до тех пор, пока не будет достигнуто значение модельного времени, заданное для этой модели в качестве одного из входных данных.

1.2.2. Элементы процедуры решения

События. При моделировании систем массового обслуживания совершаются некоторые события. Все события в системе должны быть каким-либо образом зафиксированы и должно быть учтено их воздействие на текущее состояние системы. Кроме того, необходимо определить, как нужно корректировать состояние системы в связи с воздействием на нее этих событий. События разделяются на две категории:

1. основное событие – это такое событие, время возникновения которого можно запланировать заранее. Это, например приход заявки, начало обслуживания и завершение обслуживания.
2. Вспомогательное событие – это событие, время возникновения которого невозможно запланировать заранее. Эти события возникают тогда же, когда и основные, но являются зависимыми, возникающими как следствие основных.

Таймер модельного времени. Так как работа модели связана с последовательным возникновением событий, то вполне естественно использовать понятие "Таймер Модельного Времени" в качестве одного из элементов модели системы. Для этого вводят специальную переменную и используют ее для фиксации текущего времени работы модели.

Когда начинается моделирование, таймер модельного времени обычно устанавливают на нулевое значение. Разработчик сам решает вопрос о том, какое значение реального времени принять за точку отсчета. Например, началу отсчета может соответствовать 6 ч. утра первого моделируемого дня. Разработчик также должен решить вопрос о выборе величины единицы времени. Единицей времени может быть 1 с, 5 с, 1 мин,

20 мин или 1 ч. Когда единица времени выбрана, все значения времени, получаемые при моделировании или входящие в модель, должны быть выражены через эту единицу.

На практике значения модельного времени должны быть достаточно малыми по сравнению с реальными промежутками времени, протекающими в моделируемой системе. В данной системе обычно выбирают единицу времени, равную 1 мин.

Если при моделировании некоторой системы при текущем значении модельного времени ее состояние изменилось, то нужно увеличить значение таймера. Чтобы определить, на какую величину должно быть увеличено значение таймера, используют один из двух методов:

1. Концепция фиксированного приращения значений таймера.

При таком подходе увеличивают значение таймера ровно на одну единицу времени. Затем нужно проверить состояния системы и определить те из запланированных событий, которые должны произойти при новом значении таймера. Если таковые имеются, то необходимо выполнить операции, реализующие соответствующие события, снова изменить значение таймера на одну единицу времени и т. д. Если проверка покажет, что для нового значения таймера не запланировано ни одного события, то произойдет передвижение таймера непосредственно к следующему значению.

2. Концепция переменного приращения значений таймера.

В этом случае условием, вызывающем приращение таймера, является наступление времени "близкого события". Близкое событие – это то событие, возникновение которого запланировано на момент времени, равный следующему ближайшему значению таймера модельного времени. Колебания приращения таймера от случая к случаю объясняют выражение "переменное приращение времени".

Преимущественней использовать концепцию переменного приращения значений таймера, т. к. при нем можно избежать обработки в промежуточные моменты времени, когда не планируется выполнение никаких событий.

Если список моментов времени отсортирован в порядке возрастания, то никакого поиска не требуется. Таймер просто установится в нужное значение в соответствии с первой позицией списка. Затем таймеру должно быть задано его значение, и управление должно быть передано в ту часть модели, где обслуживается событие, о котором идет речь.

Для выполнения курсовой работы рекомендуется использование языка GPSS.

2. Описание концептуальной модели

Наиболее рационально строить модель функционирования системы по блочному принципу. При этом могут быть выделены три автономные группы блоков такой модели.

Блоки первой группы представляют собой имитатор воздействий внешней среды $\boldsymbol{E}$ на систему $\boldsymbol{S}$.

Блоки второй группы являются собственно моделью процесса функционирования исследуемой системы $\boldsymbol{S}$.

Блоки третьей группы – вспомогательными и служат для машинной реализации блоков двух первых групп, а также для фиксации и обработки результатов моделирования.

Элементы системы $\boldsymbol{S}$ группируются в блоки $\boldsymbol{S_I}$ $\boldsymbol{S_{II}}$, $\boldsymbol{S_{III}}$, отражающие процесс функционирования исследуемой системы. Построенная блочная модель процесса функционирования исследуемой системы $\boldsymbol{S}$ предназначена для анализа характеристик этого процесса, который может быть проведен при машинной реализации полученной модели (рис. 3).

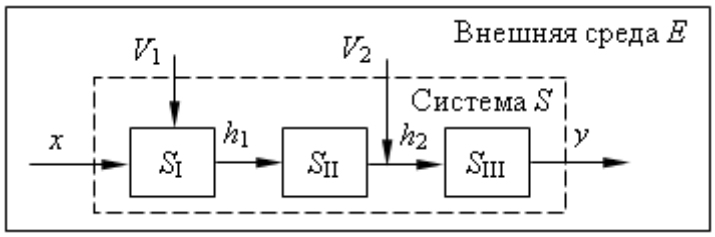


Рис. 3. Исследуемая система

Математическая модель представляет собой совокупность соотношений (например, уравнений, логических условий, операторов), определяющих характеристики процесса функционирования системы ***S*** в зависимости от структуры системы, алгоритмов поведения, параметров системы, воздействий внешней среды ***E***, начальных условий и времени.

Математическая модель является результатом формализации процесса функционирования исследуемой системы.

Однако на практике получение модели достаточно простого вида для больших систем чаще всего невозможно, поэтому обычно процесс функционирования системы ***S*** разбивают на ряд элементарных подпроцессов. При этом необходимо так проводить разбиение на подпроцессы, чтобы построение моделей отдельных подпроцессов было элементарно и не вызывало трудностей при формализации.

Одним из основных подэтапов является описание концептуальной модели системы. На этом подэтапе построения модели системы:

а) описывается концептуальная модель $\boldsymbol{M_x}$ в абстрактных терминах и понятиях;

б) дается описание модели с использованием типовых математических схем;

в) принимаются окончательно гипотезы и предположения;

г) обосновывается выбор процедуры аппроксимации реальных процессов при построении модели.

Покажем построение концептуальной модели на примере фрагмента сети передачи данных (СПД) [4].

Структурная схема модели процесса взаимодействия двух узлов коммутации (УК1 и УК2) через дискретный канал связи (ДКС) в символике *Q*-схем показана на рис. 4, где приняты следующие обозначения: И – источник; К – канал; Н – накопитель.

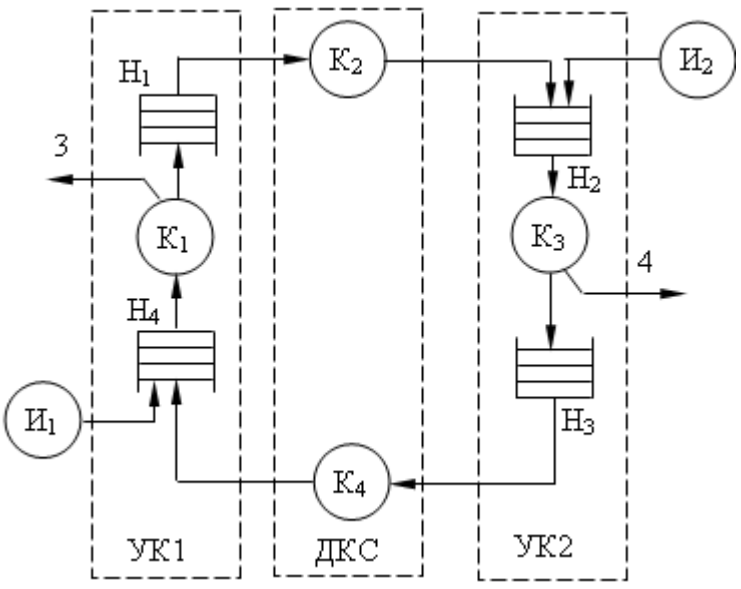


Рис. 4. Структурная схема модели процесса взаимодействия двух узлов коммутации

С использованием введенных представлений и обозначений опишем процесс функционирования фрагмента СПД. Поступление пакетов данных в моделируемый фрагмент СПД на входы имитируется источниками $И_1$ и $И_2$. Пакеты буферируются накопителями $Н_4$ и $Н_2$, где ожидают освобождения каналов в УК1 и УК2 соответственно. После обслуживания каналами $К_1$ и $К_3$, т. е. после обработки ЦП УК1 и УК2 соответственно, пакеты поступают в выходные накопители $Н_1$ и $Н_3$ этих узлов.

Далее, в порядке очереди, копии пакетов обслуживаются каналами $К_2$ и $К_4$, имитирующими процесс передачи по ДКС. При приеме копий пакета без ошибки, т. е. их поступлении в $Н_2$ и $Н_4$, формируется подтверждение приема, которое в виде короткого пакета поступает в соответствующий выходной для данного узла накопитель $Н_1$ и $Н_3$ для передачи на другой канал УК, т. е. снова реализуется обслуживание каналами $К_2$ и $К_4$. После подтверждения в узле-источнике правильного приема уничтожается пакет, хранившийся в $Н_1$ и $Н_3$. Выходам соответствуют точки 3 и 4 на структурной схеме модели процесса взаимодействия двух узлов коммутации СПД (рис. 4).

3. Блок-диаграмма в пакете GPSS

В пакете GPSS для представления моделируемой системы ***S*** в виде машинной модели используется язык блок-диаграмм. Блок-диаграммой в пакете GPSS называется графическое представление операций, происходящих в моделируемой системе ***S***. В этом случае блок-диаграмма описывает взаимодействия, происходящие внутри моделируемой системы ***S*** в процессе ее функционирования. Вводимый набор блоков для блок-диаграмм однозначно определяет наборы операторов языка, осуществляющих описание структуры моделируемой системы ***S***, и логических правил, определяющих ее функционирование.

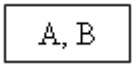
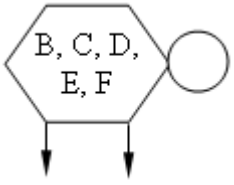
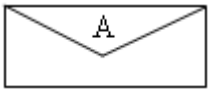
В блок-диаграммах GPSS блоки представляют собой выполняемые над динамическими объектами операции, а стрелки между блоками отражают маршруты передвижения данных объектов по системе. Альтернативные ситуации отражаются более чем одной стрелкой, выходящей из блока.

Таким образом, процесс создания модели на языке блок-диаграмм GPSS сводится к декомпозиции исходной системы ***S*** до уровня элементарных операций, используемых в пакете GPSS, формированию фиксированной схемы, отражающей последовательность элементарных операций, выполняемых над динамическими объектами, и определению набора логико-вероятностных правил продвижения потоков объектов по имеющейся схеме.

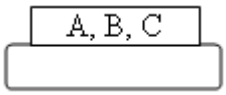
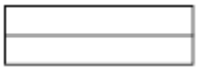
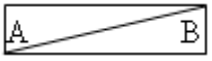
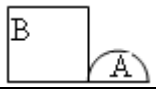
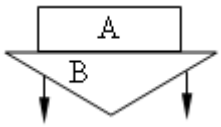
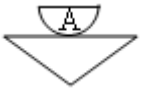
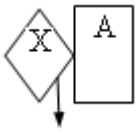
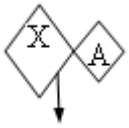
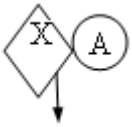
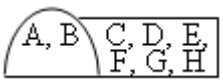
Построение блок-диаграмм GPSS предполагает знакомство программиста с набором операторов пакета GPSS, который однозначно соответствует набору блоков для описания блок-диаграмм, поэтому построение блок-диаграммы не является самоцелью, а лишь промежуточным этапом при построении имитационной модели исследуемой системы ***S*** с использованием операторов пакета GPSS.

Основные условные обозначения, используемые на блок-диаграммах GPSS, представлены в табл. 3.

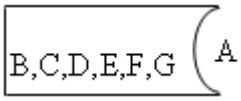
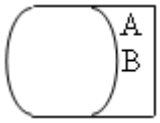
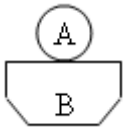
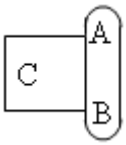
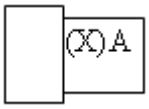
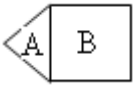
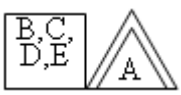
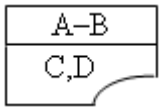
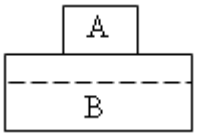
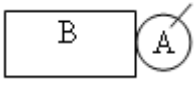
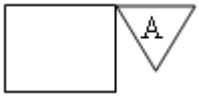
Табл. 3. Условные обозначения на блок-диаграммах GPSS

Имя блока	Обозначение блока	Назначение блока
ADVANCE		Задерживает транзакт на время A±B, если B – const, или A·B, если B – функция
ALTER		Изменяет атрибуты членов группы A
ASSEMBLE		Собирает A транзактов одного ансамбля, пропускает в следующий блок первый транзакт, остальные уничтожает

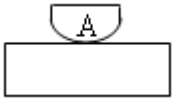
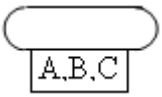
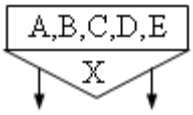
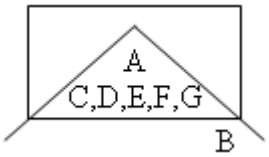
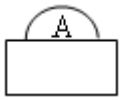
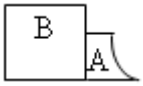
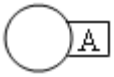
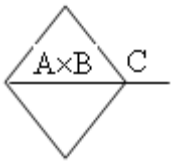
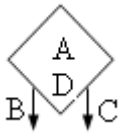
Продолжение табл. 3

Имя блока	Обозначение блока	Назначение блока
ASSIGN		Присваивает параметру А входящего транзакта значение В, модифицированное значением С
BUFFER		Прерывает обработку транзакта и возвращает симулятор к началу списка текущих событий
CHANGE		Заменяет в процессе счета блок с номером А на блок с номером В
DEPART		Обеспечивает освобождение в очереди А В единиц
ENTER		Обеспечивает вхождение транзакта в накопитель А с занятием В единиц памяти
EXAMINE		Изменяет маршрут движения в зависимости от состояния членов группы А
EXECUTE		Выполняет операции блока с номером А.
FAVAIL		Объявляет группу устройств А доступным
FUNAVAIL		Объявляет группу устройств А недоступным
GATE (X) LS LR		Проверяет условие нахождения логического ключа А в состоянии Х
GATE (X) I NI U NU		Проверяет условие нахождения устройства А в состоянии Х
GATE (X) SE SF SNE SNE		Проверяет условие нахождения накопителя А в состоянии Х
GATHER		Собирает А транзактов одного ансамбля и пропускает их одновременно в следующий блок
GENERATE		Генерирует транзакты через А единиц времени, модифицированных В с задержкой С, D транзактов, с приоритетом Е, форматом F

Продолжение табл. 3

Имя блока	Обозначение блока	Назначение блока
HELP		Объединяет модули ПМДС с модулями на языках Ассемблера, Паскаля и т.д.
INDEX		Записывает сумму значений параметра А и величины В в параметр 1 типа А
JOIN		Включает в группу А транзакт или числовое значение В
LEAVE		Освобождает в памяти А В единиц памяти
LINK		Удаляет транзакт из списка текущих событий и помещает в список пользователя
LOGIC (X) S R I		Устанавливает логический ключ А в состояние Х
LOOP		Осуществляет повторение А раз группы блоков от адреса В до данного блока
MARK		Осуществляет отметку времени в параметре А
MATCH		Синхронизирует движение транзактов по блок-диаграмме совместно с блоком MATCH с меткой А
PREEMPT		Выполняет приоритетную обработку в устройстве А
PRINT		Осуществляет вывод статистики С в пределах от А до В
PRIORITY		Присваивает входящему транзакту приоритет А
QUEUE		Обеспечивает занятие В мест в очереди А
RELEASE		Освобождает устройство с номером А

Продолжение табл. 3

Имя блока	Обозначение блока	Назначение блока
RETURN		Снимает прерывание с устройства A
SAVAIL		Объявляет группу памятей A доступными
SAVEVALUE		Сохраняет заданное значение B в ячейке A
SCAN		Обрабатывает атрибуты членов группы A
SEIZE		Занимает устройство с номером A
SPLIT		Генерирует A копий входящего транзакта и направляет их по адресу B; основной транзакт переходит в следующий блок
SUNAVAIL		Объявляет группу памятей A недоступными
TABULATE		Табулирует значения входящих транзактов в таблице A
TERMINATE		Уничтожает A транзактов
TRACE		Прослеживает движение транзактов
TEST (X) E NE GE LE G L X		Проверяет соотношение × между A и B и направляет входящий транзакт в следующий блок при выполнении или по адресу C при невыполнении соотношения
TRANSFER		Изменяет направление движения транзактов согласно режиму A

4. Общие сведения о GPSS/PC

Язык GPSS – это язык декларативного типа, построенный по принципу объектно-

ориентированного языка. Основными элементами этого языка являются транзакты и блоки, которые отображают соответственно динамические и статические объекты моделируемой системы.

Предназначение объектов системы различно. Выбор объектов в конкретной модели зависит от характеристик моделируемой системы. Каждый объект имеет некоторое число свойств, названых в GPSS стандартными числовыми атрибутами (СЧА). Часть СЧА доступна пользователю только для чтения, а на значение других он может влиять, используя соответствующие блоки.

4.1 Блоки и транзакты

Каждая GPSS-модель обязательно должна содержать такие объекты, как блоки и транзакты.

В GPSS концепция передачи управления от блока к блоку имеет специфические особенности. Последовательность блоков GPSS-модели показывает направления, в которых перемещаются элементы. Каждый такой элемент называется *транзактом.* Транзакты – это динамические элементы GPSS-модели.

Блоки языка GPSS представляют собой подпрограммы, написанные на макроассемблере или на языке Си, и содержат набор параметров (операндов) для обращения к ним. Как и во всех языках моделирования в GPSS существует внутренний механизм передачи управления, который реализуется в модельном времени, что дает возможность отобразить динамические процессы в реальных системах. Передача управления от блока к блоку в GPSS-программах реализуется с помощью движения транзактов в модельном времени. Обращение к подпрограммам блоков происходит через движение транзактов.

Содержательное значение транзактов определяет разработчик модели. Именно он устанавливает аналогию между транзактами и реальными динамическими элементами моделируемой системы.

Такая аналогия никогда не указывается транслятору GPSS, она остается в воображении разработчика моделей. В табл. 4 приведены примеры аналогий между транзактами и элементами реальных систем.

Табл. 4. Примеры аналогий между транзактами и элементами реальных систем

Система	Элементы систем, которые моделируются транзактами
Телефонная станция	Абонент
Компьютерный класс	Пользователь
Склад	Заявка

С точки зрения программы – *транзакт это структура данных,* которая содержит такие поля: имя или номер транзакта; время появления транзакта; текущее модельное время; номер блока, в котором находится транзакт; номер блока, куда он продвигается; момент времени начала продвижения; приоритет транзакта; параметры транзакта: P1, P2, ...

В языке GPSS все транзакты нумеруются по мере их появления в модели. Параметры транзактов отображают свойства моделируемого динамического объекта. Например, если моделируется движение автомобилей на участке дороги, то параметрами транзакта (автомобиля) в зависимости от целей моделирования могут быть скорость, тормозной путь, габариты и др.

Каждый транзакт занимает некоторый объем памяти ЭВМ. После того, как он закончит свое движение по блокам модели, его необходимо уничтожать для освобождения памяти, чтобы избежать ее переполнения. Поскольку транслятору не известно

сколько транзактов одновременно будет находиться в модели, то память под транзакты выделяется динамически.

Таким образом, при начале моделирования в GPSS-модели не существует ни одного транзакта. В процессе моделирования транзакты входят в модель в определенные моменты времени, соответствующие логике функционирования моделируемой системы. Таким же образом транзакты покидают модель в зависимости от специфики моделирования. В общем случае в модели существует несколько транзактов, но в каждый момент времени движется только один из них.

Если транзакт начал свое движение, он передвигается от блока к блоку по пути, указанному блок-схемой (логикой работы модели). В тот момент, когда транзакт входит в блок, вызывается соответствующая этому блоку подпрограмма. Далее транзакт (в общем случае) пытается войти в следующий блок. Его перемещение продолжается до тех пор, пока не выполнится одно из таких возможных условий:

1. Транзакт входит в блок, функцией которого является задержка транзакта на определенное время.

2. Транзакт входит в блок, функцией которого является удаление транзакта из модели.

3. В соответствии с логикой модели транзакт пытается войти в следующий блок, но блок не принимает этот транзакт. В этом случае транзакт остается в том блоке, в котором в данное время находится, но позже будет повторять попытки войти в следующий блок. Когда условия в модели изменятся, одна из таких попыток может быть успешной. После этого транзакт продолжит свое перемещение по модели.

Если выполняется одно из указанных условий, транзакт остается на месте и в модели делается попытка перемещения другого транзакта.

*Объекты типа «ресурсы»***.** Аналогами обслуживающих устройств реальных систем в GPSS являются объекты типа «ресурсы». К объектам этого типа относятся устройства, многоканальные устройства и логические ключи.

Как и в каждом объектно-ориентированном языке в GPSS каждый объект имеет *свойства и методы,* которые изменяют эти свойства. В GPSS свойства объектов называют *стандартными числовыми атрибутами* (СЧА).

Устройство (одноканальное устройство, прибор) представляет собой ресурс, который в любой момент времени может быть занят только одним транзактом. Интерпретатор автоматически вычисляет такие его СЧА, как общее время занятости устройства, число транзактов, который занимали устройство, коэффициент использования устройства, среднее время занятости устройства одним транзактом и т. п.

Многоканальные устройства (МКУ) (несколько параллельных одинаковых устройств) представляют собой объекты типа «ресурсы» для параллельной обработки. Они могут быть использованы несколькими транзактами одновременно. Пользователь определяет емкость каждого МКУ, который используется в модели, а интерпретатор ведет учет числа устройств, занятых в каждый момент времени. Интерпретатор также автоматически подсчитывает такие СЧА: число транзактов, которые вошли в МКУ; среднее число каналов, занятых одним транзактом; среднее время нахождения транзакта в устройстве и др.

Некоторые события в системе могут заблокировать или изменить движение транзактов. Например, кассир кинотеатра, идя на обед, ставит табличку «В другое окно», и все следующие клиенты на протяжении обеда обращаются в другую кассу. Для моделирования этих ситуаций введены логические ключи. Транзакт может устанавливать эти ключи в положение «Включено» или «Выключено». Через некоторое время состояние ключа может быть использовано другими транзактами для выбора одного из двух возможных путей движения или ожидания момента изменения состояния ключа. Состоя-

ние ключа может быть изменено любым транзактом.

Переменные. *Арифметические переменные* позволяют вычислять арифметические выражения, которые состоят из операций над СЧА объектов. В выражениях могут быть использованы функции (библиотечные или пользовательские). *Булевы переменные* позволяют пользователю одновременно проверять несколько условий, исходя из состояния объектов или значений СЧА.

Функции. Используя *функции*, пользователь может задавать непрерывную или дискретную функциональную зависимость между аргументом функции и ее значением. Функции в GPSS задаются табличным способом с помощью операторов описания функций.

Ячейки и матрицы сохраняемых величин. Ячейки сохраняемых величин и матрицы используются для хранения некоторой пользовательской числовой информации, запись в эти объекты выполняют транзакты. Записанную в этих объектах информацию может считывать любой транзакт. Таким образом, эти объекты являются глобальными и доступны из любой части модели.

Очереди. В любой системе движение потока транзактов может быть задержано из-за недоступности ресурсов (например, необходимые устройства или МКУ уже заняты). В этом случае задержанные транзакты становятся в *очередь* – еще один тип объектов GPSS. Учет этих очередей составляет одну из основных функций интерпретатора.

Пользователь может специально определить точки модели, в которых необходимо собирать статистику об очередях, то есть установить *регистраторы очереди*. Тогда интерпретатор будет автоматически собирать статистику об очередях (длину очереди, среднее время нахождения в очереди и т.п.). Вся эта информация является СЧА и доступна пользователю в процессе моделирования.

Интерпретатором автоматически поддерживается дисциплина обслуживания очереди FIFO («первым пришел – первым обслужился»), и пользователь может получить стандартную статистическую информацию только об этих очередях. Если у пользователя возникает необходимость организовать очередь из транзактов с другой дисциплиной обслуживания (например, LIFO), то для этого используются *списки пользователей*. Эти списки также помогают осуществлять синхронизацию движения разных транзактов по модели.

Таблицы. Объект «*таблица*» предназначен для сбора статистики о случайных величинах, заданных пользователем. Таблица состоит из частотных классов, в которые заносится число попаданий конкретной величины (некоторого СЧА). Для каждой таблицы вычисляется математическое ожидание и среднеквадратическое отклонение.

4.2. ЧАСЫ модельного времени

Разные события реальных систем происходят в течение некоторого периода времени. Например, покупатели приходят в магазин, когда подходит их очередь, они попадают на обслуживание. Когда покупки сделаны, покупатели покидают магазин. Если все эти события представить в модели, то их возникновение должно происходить на фоне модельного времени. Интерпретатор автоматически обслуживает часы модельного времени.

В момент начала моделирования интерпретатор планирует появление первого транзакта. После этого часы модельного времени устанавливаются на значение времени, которое соответствует моменту появления первого транзакта в модели. Этот транзакт (и другие, если они приходят в этот же момент времени) входит в модель. Далее он передвигается через все возможные блоки модели, которые ему встречаются. События, которые возникают вследствие перемещения транзакта через блоки, планируются на

дальнейшие моменты времени. Естественно, что в этот первый отмеченный момент времени ничего больше в системе не происходит. Интерпретатор GPSS продвигает дальше значения ЧАСОВ к тому значению времени, на которое запланировано следующее ближайшее событие. Если во второй, отмеченный ЧАСАМИ момент времени, нет транзактов, которые нужно перемещать, ЧАСЫ снова продвигаются вперед и т.д. Именно так, от события к событию, и происходит смена модельного времени.

Особенности часов GPSS:

1. Часы в OP88 регистрируют целые значения (за исключением языка GPSS World, где время может иметь действительные значения), то есть события могут появляться только в целые моменты времени. Это сделано с целью ускорения процесса моделирования, поскольку целочисленная арифметика выполняется процессором ЭВМ быстрее и требует меньше памяти.

2. Единица модельного времени определяется разработчиком. Эту единицу времени интерпретатору не сообщают. Значение принятой единицы модельного времени выражают в неявном виде в форме временных данных модели. Так, если все данные выражены в минутах, то единицей времени будет минута, то есть масштаб времени в модели будет такой: одна единица модельного времени равна одной минуте реального времени. Если все данные выражены в миллисекундах, то единицей модельного времени будет миллисекунда. Разработчик может задавать такую единицу времени, которая ему удобна для того, чтобы правильно отобразить события реальной системы в модели.

3. Система GPSS является интерпретатором «следующего события». Иначе говоря, после того, как модель полностью скорректирована в данный момент дискретно изменяющегося времени, ЧАСЫ перемещаются к следующему моменту времени, на который запланировано следующее событие. Таким образом, ЧАСЫ модельного времени продвигаются от одного события к другому.

4.3. Типы операторов

Операторы OP88 делятся на три типа:

1) блоки;

2) операторы описания данных;

3) команды GPSS.

Общие сведения о формате операторов GPSS. В GPSS для ссылки на числа, блоки и объекты используются *имена* (идентификаторы). Имя представляет собой алфавитно-цифровую последовательность длиной до 20 символов в GPSS/PC и до 250 символов в GPSS World, которая начинается с буквы. Допускается использование символов только латинского алфавита, цифр и знака подчеркивания.

Формат GPSS-блоков такой:

[Номер строки] [< Метка >] < Операция > < Операнды > <; Комментарии >

Номер строки. Обязательное поле для GPSS/PC (в GPSS World – игнорируется). Начинается с первой позиции строки. Представляет собой десятичное число.

Метка (имя блока). Содержимым поля является имя - последовательность символов, начинающаяся с буквы. В некоторых операторах это поле является обязательным.

Операция. Операциями блоков являются глаголы, которые описывают основные функциональные назначения блоков. Каждый из блоков характеризируется своим собственным предписанным ему глаголом.

Операнды. Блоки могут иметь операнды. Операнды блоков задают информацию, специфичную для действия данного блока. Число операндов блока зависит от типа блока. В блоках не может использоваться больше семи операндов. Операнды в общем слу-

чае обозначаются символами: **A, B, C, D, E, F, G**. Значения операндов определяются типом блока. Одни операнды некоторых блоков должны быть определены всегда, а другие могут задаваться или не задаваться (т.е. являются необязательными). Операнды следуют один за другим и отделяются запятыми или одним пробелом. Если операнд опущен, то вместо него ставится запятая. Между операндами не должно быть более одного пробела, так как это будет означать, что операнды закончились и интерпретатор прекращает чтение строки.

Комментарии. Необязательное поле. Комментарии отделяются от поля операндов символом «;». Допускается запись комментария с начала строки. В этом случае в первой позиции строки ставится символ «;» или «*». В GPSS/PC допускаются комментарии с использованием заглавных или строчных букв только латинского алфавита, в GPSS World также допускается использование символов кириллицы.

Строка описания блока может содержать до 79 символов в GPSS/PC и до 250 символов в GPSS World. При описании форматов квадратные скобки [ ] указывают на необязательность поля.

Именами и метками не могут быть названия или начальные символы в названии блоков, операторов, команд и СЧА. Во избежание конфликтов с ключевыми словами рекомендуется в именах использовать символ подчеркивания.

4.4. Внесение транзактов в модель. Блок GENERATE

Блок **GENERATE (ГЕНЕРИРОВАТЬ)** – это блок, через который транзакты входят в модель. Не существует ограничений на количество разных блоков **GENERATE** в одной модели.

Интервал времени между последовательными появлениями транзактов из блока **GENERATE** называют *интервалом поступления*. Когда транзакт входит в модель через блок **GENERATE,** интерпретатор планирует время поступления следующего транзакта путем розыгрыша случайного числа с соответствующим распределением интервалов поступления на время, равное текущему значению ЧАСОВ плюс разыгранное значение. При достижении этого значения модельного времени следующий транзакт вводится в модель через блок **GENERATE** и т.д.

Разработчик должен задать функцию распределения интервалов поступления транзактов в блоке **GENERATE.**

Все возможные виды случайных распределений интервалов поступления транзактов в GPSS делятся на равномерное распределение и другие виды распределений. В нашем случае специально рассматривают самое простое из всех случайных нетривиальных распределений – равномерное распределение. Использование других видов распределений требует задания функций.

Формат блока (значение операндов приведено в табл. 5):

GENERATE [A],[B],[C],[D],[E]

Табл. 5. Значение операндов

Операнд	Значение	Значение по умолчанию
A	Средний интервал времени (число, СЧА)	0
B	Половина поля допуска равномерно распределенного интервала (число, СЧА)	0
C	Смещение интервалов	Смещение отсутствует
B	Ограничитель транзактов	∞
E	Уровень приоритета транзакта. Возможные значения 0–127	0

Значение операндов:

A – среднее значение интервала поступления;

B – величина разброса возможных значений относительно среднего значения. (Если операнд **B** не задается, то интервал времени поступления - детерминированная величина);

C – момент времени, в который в блоке **GENERATE** должен появиться первый транзакт. (После этого первого прихода все остальные приходы транзактов возникают в соответствии с распределением, заданным операндами **A** и **B**);

D – ограничитель общего числа транзактов, которое может войти в модель через данный блок **GENERATE** на протяжении времени моделирования. (Если это число достигнуто, данный блок **GENERATE** перестает быть активным);

E – уровень или класс приоритета каждого из транзактов, которые вводятся в модель через данный блок **GENERATE.** (Всего существует 128 разных уровней, которые задаются с помощью чисел от 0 до 127. Чем больше число, тем выше приоритет).

1. Транзакты не могут входить в блок **GENERATE,** так как он сам их генерирует.
2. Если в модели GPSS/PC встречаются подряд два или больше блоков **GENERATE,** то последний блок переопределяет операнды предыдущих блоков. В GPSS World транслятор выдает ошибку.
3. Операнды не могут быть отрицательными числами.

Операнды **A, B, C** целочисленные (в GPSS World могут быть действительными числами).

4.5. Удаление транзактов из модели. Блок TERMINATE

Транзакты удаляются из модели, попадая в блок **TERMINATE (ЗАВЕРШИТЬ).** В этот момент освобождается память, выделенная под транзакт. Эти блоки всегда позволяют выйти всем транзактам, которые пытаются это сделать. В модели может быть любое количество блоков **TERMINATE.**

Формат блока:

TERMINATE [A]

Операнд **A** является величиной уменьшения специального счетчика, который называется *счетчиком завершения*. Этот операнд задает величину, которая вычитается из счетчика каждый раз, когда транзакт входит в блок **TERMINATE.** По умолчанию **A** = 0. Вход транзакта в блок **TERMINATE** с нулевым значением операнда **A** не вызывает уменьшения счетчика завершения.

Счетчик завершения – это ячейка в памяти ЭВМ, которая хранит целое положительное число. Начальное значение этого счетчика устанавливается в начале моделирования. Оно равняется значению операнда **A** команды **START** (НАЧАТЬ). В процессе моделирования транзакты попадают в блок **TERMINATE** и, таким образом, уменьшают значение счетчика на величину операнда **A.** Моделирование заканчивается, когда значение счетчика становится равным нулю или отрицательному числу.

1. В модели может быть много блоков **TERMINATE,** но счетчик завершения – один, с начальным значением, указанным в команде **START.**
2. Не путать ограничитель транзактов в блоке **GENERATE** и счетчик завершения. Ограничитель задает число транзактов, которые войдут в модель, а счетчик – число транзактов, которые выйдут из модели. По окончании моделирования транзакты могут оставаться в модели.

Интерпретатор начинает моделирование по команде **START.**

Ее формат:

START A,[B],[C],[D]

В операнде **A** задается начальное значение счетчика завершения.

Управление продолжительностью процесса моделирования. В языке GPSS продолжительностью процесса моделирования можно управлять двумя способами:

1) завершать моделирование после того, как модель покинет заданное число транзактов определенного типа;

2) завершать моделирование по истечению заданного интервала времени.

Первый способ:

1. В команде **START** операнду **A** присваивается значение заданного числа транзактов.

2. Во всех блоках **TERMINATE,** через которые транзакты заданного типа покидают модель, операнду **A** присваивается значение «1» или другое, отличное от нуля (соответственно содержательному значению транзактов).

3. Во все других блоках **TERMINATE** используется значение операнда **A** по умолчанию (**A** = 0). Значение счетчика завершения не будет зависеть от этих блоков.

Первый способ позволяет закончить моделирование, когда через модель пройдет заданное количество транзактов, например 1000:

```
GENERATE        40,5
TERMINATE
START   1000
```

Второй способ:

Пусть разработчик выбрал за единицу модельного времени 1 мин и хочет смоделировать поведение системы на протяжении 8 часов. Это можно сделать таким образом:

1. Ввести в модель таймер-сегмент, состоящий из двух блоков:

```
GENERATE    480
TERMINATE   I
```

2. Во всех других блоках **TERMINATE** в модели использовать значение операнда **A** по умолчанию (**A** = 0). Это означает, что прекращение моделирования, определяемое счетчиком завершения, не будет зависеть от других блоков **TERMINATE.**

3. В команде **START** операнд **A** должен равняться единице.

Таким образом, в процессе моделирования завершение движения транзактов в других блоках **TERMINATE** не влияет на счетчик завершения. В момент времени 480 транзакт выйдет из блока **GENERATE** и сразу же перейдет в блок **TERMINATE.** Счетчик завершения уменьшится на единицу, и интерпретатор завершит моделирование.

4.6. Элементы, отображающие одноканальные обслуживающие устройства

Рассмотрим элементы, которые используются для представления обслуживания. Аналогами обслуживающих элементов могут быть люди, механизмы, линии связи и другие объекты реальных систем. В GPSS такие объекты моделируются с помощью устройств, МКУ, логических ключей.

Устройство характеризируется двумя основными свойствами:

1. Каждое устройство в любой момент времени может обслуживать только один транзакт. Если в процессе обслуживания появляется новый транзакт, то он должен:

- либо подождать своей очереди,
- либо направиться в другое место,

- либо, если вновь пришедший транзакт имеет больший приоритет, устройство прерывает текущее обслуживание и начинает обслуживать новый транзакт.

2. Когда транзакт поступает в устройство, он должен пробыть там необходимое для обслуживания время.

Всем устройствам необходимо задавать имена. Они могут быть или числовыми (числа должны быть положительными целыми), или символьными. Во время трансляции символьным именам сам транслятор присваивает числовые значения.

Для того чтобы использовать одноканальное обслуживающее устройство (прибор), транзакту необходимо выполнить следующие шаги.

Первый шаг. Ждать своей очереди, если это необходимо. Ожидание длится в течение некоторого интервала времени.

Второй шаг. Когда подходит очередь, занять устройство. Событие «занятие устройства» происходит в некоторый момент модельного времени.

Третий шаг. Устройство находится в состоянии занятости до тех пор, пока не закончится обслуживание. Для обслуживания необходим некоторый интервал времени.

Четвертый шаг. Когда обслуживание закончится, освободить устройство. Событие «освобождение устройства» происходит в некоторый момент модельного времени.

Эта последовательность шагов выполняется GPSS при моделировании использования устройства. Второй и четвертый шаги реализуются блоками **SEIZE (ЗАНЯТЬ)** и **RELEASE (ОСВОБОДИТЬ).**

Формат блока (значение операнда приведено в табл. 6):

SEIZE A

Табл. 6. Значение операнда

Операнд	Значение	Результат по умолчанию
A	Имя (символьное или числовое) занимаемого устройства	Ошибка

Этот блок имеет следующие свойства:

1. Если в текущий момент времени устройство используется, то транзакт не может войти в блок и должен ожидать своей очереди.

2. Если устройство свободно, транзакт может войти в блок. Вход транзакта в блок вызывает выполнение подпрограммы обработки этого блока. Состояние устройства изменяется со **СВОБОДНОЕ** на **ЗАНЯТОЕ**.

Предварительного объявления устройства в модели не требуется, так как тот факт, что блок **SEIZE** используется, свидетельствует о существовании данного устройства.

Предназначением блока **RELEASE** является изменение состояния ранее занятого устройства с **ЗАНЯТОГО** на **СВОБОДНОЕ**. Блок **RELEASE** никогда не запрещает вход транзакта.

Формат блока (значение операнда приведено в табл. 7):

RELEASE A

Табл. 7. Значение операнда

Операнд	Значение	Результат по умолчанию
A	Имя (символьное или числовое) освобождаемого устройства	Ошибка

В то время как транзакты находятся в модели временно, устройства, используемые в модели, существуют в ней в течение всего периода моделирования.

Статистическая информация о работе устройства при моделировании собирается автоматически.

Если в модели используются объекты типа «устройство», то в файле стандартной статистики будет представлена информация об использованных устройствах (табл. 8).

Табл. 8. Информация об использованных устройствах

FACILITI Номер или имя устройства	**ENTRIES** Количество входов	**UTIL** Коэффициент использования	**AVE, TIME** Среднее время преобразования транзакта в устройстве	**AVAIL** Состояние готовности
1	50	0,07	74,06	
2	51	0,10	100,29	

OWNER Номер последнего транзакта, занявшего устройства	**PEND** Количество прерванных в устройстве транзактов	**INTER** Количество прерывающих устройств транзактов	**RETRY** Количество транзактов, ожидающих специальных условий	**DELAY** Количество транзактов, ожидающих занятия устройства

Статистику работы устройств в процессе моделирования можно наблюдать в окне устройств для GPSS/PC (перейдя в это окно с помощью клавиш [ALT+F]) или в окне Facilities Window для GPSS World.

1. После блока **SEIZE** может сразу же следовать другой блок **SEIZE,** если транзакт должен одновременно занять два или более устройств (например, рабочего и инструмент).

2.Транзакт не может освободить устройство, которое он не занимал.

4.7. Реализация задержки во времени. Блок ADVANCE

Перевод с английского языка блока **ADVANCE (ЗАДЕРЖАТЬ)** – продвигать, а не задерживать. Этот блок действительно продвигает ЧАСЫ модельного времени на некоторое значение, но фактически он осуществляет задержку продвижения транзакта в течение некоторого интервала времени. Обычно этот интервал задается случайной величиной.

В GPSS возможны следующие варианты распределения времени обслуживания:

1) детерминированное (постоянное);

2) равномерное распределение;

3) другие распределения.

Как и при использовании блока **GENERATE** особо рассматривается равномерное распределение случайных величин.

Формат блока (значение операндов приведено в табл. 9):

ADVANCE A[,B]

Табл. 9. Значения операндов

Операнд	Значение	Результат по умолчанию
A	Среднее время задержки на обслуживание (число, СЧА)	0
B	Половина поля допуска равномерно распределенного времени задержки (число, СЧА)	0

Блок никогда не препятствует входу транзакта. Любое число транзактов может находиться в этом блоке одновременно. Когда транзакт попадает в такой блок, выпол-

няется соответствующая подпрограмма и вычисляется время пребывания в нем транзакта. Вновь прибывший транзакт никак не влияет на уже находящийся в блоке транзакт.

Если время пребывания в блоке равно нулю, то вместо задержки в блоке **ADVANCE** интерпретатор сразу же пытается переместить этот транзакт в следующий блок.

1. В GPSS/PC не допускаются дробные значения времени задержки. 2.Отрицательное значение задержки всегда вызывает ошибку.

Блоки **ADVANCE** можно располагать в любых местах программы, а не только между блоками **SEIZE и RELEASE.**

4.8. Переход транзакта в блок, отличный от последующего. Блок TRANSFER

В GPSS блок **TRANSFER (ПЕРЕДАТЬ)** может быть использован в девяти разных режимах. Рассмотрим три основных.

Блок TRANSFER в режиме безусловной передачи.

Его формат (значение операндов приведено в табл. 10):

TRANSFER ,B

Табл. 10. Значение операндов

Операнд	Значение	Результат по умолчанию
A	Не используется	–
B	Позиция блока, в которую должен перейти транзакт	Ошибка

Позиция блока – это номер или метка блока. Так как операнд **A** не используется, то перед операндом **B** должна стоять запятая. В режиме безусловной передачи блок **TRANSFER** не может отказывать транзакту во входе. Кстати, если транзакт входит в блок, то он сразу же пытается войти в блок **B**.

Транслятор GPSS/PC не улавливает пропущенную запятую вместо операнда **A** (например, TRANSFER LAMD). На этапе трансляции метке LAMD присваивается числовое значение, и транзакт в этом случае направляется в блок с соответствующем номером.

Статистический режим. В этом режиме осуществляется передача транзакта в один из двух блоков случайным образом.

Формат блока (значение операндов приведено в табл. 11):

TRANSFER A,[B],C

Табл. 11. Значение операндов

Операнд	Значение	Результат по умолчанию
A	Вероятность передачи транзакта в блок С, задаваемая в долях тысячи	Ошибка
B	Позиция блока, в которую должен перейти транзакт (с вероятностью 1–A)	Следующий по порядку блок
C	Позиция блока, в которую должен перейти транзакт (с вероятностью A)	Ошибка

При задании вероятности (операнд A) используется не более трех цифр, первый символ записи частоты «.» (десятичная точка), если используется действительное число, которое должно быть в пределах от 0 до 1,0 (например, 0,235). Если операнд – положительное целое число, то вероятность интерпретируется в долях тысячи.

Режим BOTH. Если в операнде **A** стоит зарезервированное слово **BOTH,** то блок **TRANSFER** работает в режиме **BOTH.**

В этом режиме входящий транзакт сначала пытается перейти к блоку, указанному в операнде **B**. Если это сделать не удается, транзакт пытается перейти в блок, указанный в операнде **C**. Если транзакт не сможет перейти ни к тому, ни к другому блоку, то он остается в блоке **TRANSFER** и при каждом просмотре списка текущих событий, будет повторять в том же порядке попытки перехода до тех пор, пока не сможет выйти из блока **TRANSFER.**

1. Не путайте метку блока **SEIZE** с именем соответствующего этому блоку устройства.

2. Если бы меткой **LL1** был помечен блок **QUEUE,** а не блок **SEIZE,** то все транзакты были бы направлены по метке **LL1,** так как в отличие от блока **SEIZE** блок **QUEUE** всегда готов принять транзакты.

4.9. Моделирование многоканальных устройств

Устройство в GPSS используют для моделирования одиночного устройства обслуживания. Два или более обслуживающих устройства, работающих параллельно, могут моделироваться в GPSS двумя или более одноканальными устройствами. Обычно это необходимо, когда отдельные устройства являются разнородными, например, имеют различную интенсивность обслуживания.

Однако очень часто параллельно работающие устройства являются одинаковыми, и GPSS предоставляет для их моделирования объект, называемый многоканальным устройством (МКУ).

Количество устройств, которое моделируется каждым из МКУ, определяется пользователем. В этом смысле употребляют термин «емкость МКУ». Эта емкость заранее должна быть определена пользователем, чтобы интерпретатор знал, сколько устройств использует данное МКУ.

Блоки ENTER (ВОЙТИ) и **LEAVE (ВЫЙТИ)**. Использование МКУ аналогично использованию одиночного устройства. Элементом, который занимает и использует МКУ, является транзакт. При моделировании МКУ события происходят в следующем порядке:

1) транзакт ожидает своей очереди, если это необходимо;

2) транзакт занимает устройство;

3) устройство осуществляет обслуживание на протяжении некоторого интервала времени;

4) транзакт освобождает устройство.

Блоки **ENTER и LEAVE** моделируют события 2 и 4.

Формат блоков (значение операндов приведено в табл. 12):

ENTER A[,B]
LEAVE A[,B]

Табл. 12. Значения операндов

Операнд	Значение	Результат по умолчанию
A	Имя МКУ	Ошибка
B	Количество занимаемых одновременно устройств	1

Когда транзакт входит в блок **ENTER,** интерпретатор выполняет следующие действия:

1) увеличивает счетчик входов МКУ на значение операнда **B**;

2) увеличивает текущее содержимое МКУ на значение операнда **B**;

3) уменьшает доступную емкость МКУ на значение операнда **B**. Когда транзакт входит в блок **LEAVE,** интерпретатор выполняет обратные действия:

1) уменьшает текущее содержимое МКУ на значение операнда **B**;

2) увеличивает доступную емкость МКУ на значение операнда **B**.

Операнду **B** можно присвоить значение, отличное от единицы. Например, пусть транзакт моделирует корабль, а МКУ – причалы в порту. В зависимости от размера корабль может занимать нескольких причалов, т.е. **B>1.**

Если в модели используются объекты типа МКУ, то в файле стандартной статистики об этих объектах будет представлена информация (табл. 13):

Табл. 13. Информация об объектах

STOR-AGE	CAP.	RE-MAIN	MIN	MAX	EN-TRIES	AVL.	AVE.C.	UTIL.	RE-TRY	DE-LAY
RPOOL	3	3	0	1	50	1	0,99	0,33	0	0

Поле **STORAGE** определяет имя или номер МКУ.

Поле **CAP.** определяет емкость МКУ, заданную оператором **STORAGE.**

Поле **REMAIN** определяет количество единиц свободной емкости МКУ в конце периода моделирования.

Поле **MIN** определяет минимальное количество используемой емкости МКУ за период моделирования.

Поле **MAX** определяет максимальное количество используемой емкости МКУ за период моделирования.

Поле **ENTRIES** определяет количество входов в МКУ за период моделирования.

Поле **AVL.** определяет состояние готовности МКУ в конце периода моделирования: 1 – МКУ готов, 0 – не готов.

Поле **AVE.C** определяет среднее значение занятой емкости за период моделирования.

Поле **UTIL.** определяет средний коэффициент использования всех устройств МКУ.

Поле **RETRY** определяет количество транзактов, ожидающих специальных условий, зависящих от состояния МКУ.

Поле **DELAY** определяет количество транзактов, ожидающих возможности входа в блок **ENTER.**

Для GPSS/PC статистику о работе МКУ можно наблюдать в окне МКУ, перейдя в это окно с помощью клавиш [ALT+S], а для GPSS World – в окне Storages Window.

Определение емкости МКУ. Все используемые в модели МКУ должны быть заранее описаны, т.е. должно быть определено количество однотипных устройств, входящих в МКУ. Для этого используется оператор **STORAGE** (**ХРАНИЛИЩЕ** или **ПАМЯТЬ**), определяющий емкость МКУ. Название **STORAGE** становится понятным, если представить себе, что МКУ это автоматизированный склад или многоэтажный гараж с определенным числом мест, которое и задает этот оператор. В таких случаях МКУ определяет не количество одинаковых устройств для обслуживания, а количество одинаковых мест для хранения.

Формат оператора задания емкости МКУ (табл. 14):

Табл. 14. Формат оператора задания емкости МКУ

Поле	Информация в поле
Метка	Символическое имя МКУ
Операция	**STORAGE**
Операнд **A**	Емкость МКУ

4.10. Переменные

Общая характеристика переменных. При построении модели системы, иногда возникает необходимость задать сложные математические или логические соотношения между атрибутами системы. Для этой цели в программе используются переменные.

В GPSS имеется три типа переменных:

1) арифметические переменные;

2) арифметические переменные с «плавающей точкой»;

3) булевы переменные.

Значение арифметических переменных может использоваться как:

1) операнд блока; в этом случае значение арифметической переменной может представлять собой:

- номер объекта (устройства, МКУ, очереди и т. п.);
- номер параметра транзакта;
- значение стандартного числового атрибута;

3) операнд **A** функции;

4) операнд **A** таблицы;

5) операнд выражения другой переменной. В выражениях арифметические переменные используют такие арифметические операции:

\+ алгебраическое сложение;

– алгебраическое вычитание;

* алгебраическое умножение;

/ алгебраическое деление (результатом операции является целая часть частного);

@ деление по модулю;

^ возведение в степень;

| деление без остатка (перед делением у обоих операндов отбрасываются дробные части, результатом операции есть целая "часть частного).

Привычно используемый для умножения во многих языках знак - «*», зарезервирован в GPSS для обозначения косвенной адресации, однако в GPSS World есть возможность в меню настройки параметров переопределить для умножения знак «*», а для косвенной адресации – «#». Косвенная адресация является мощным средством для построения компактных и гибких моделей. Ее идея заключается в том, что можно обратиться к любому объекту или СЧА через параметры транзактов. Доступ же к параметрам транзактов осуществляется через СЧА P_j, где j – номер параметра транзакта (например, 10) или **P$имя,** где *имя* – имя (идентификатор) параметра транзакта. Так как обращение к объекту возможно только через параметр транзакта, то символ **P** может опускаться. Например, выражение **Q*7** или **Q*P7** определяет текущее значение длины очереди, номер которой задан в параметре 7 транзакта. Если в седьмом параметре хранится значение 3, то это будет текущее значение длины очереди с номером 3.

В выражениях может быть задано любое число приведенных операций в различных комбинациях. Знак результата вычисляется по обычным алгебраическим правилам. Допускаются отрицательные значения переменных. Выражения анализируются слева направо. Возведение в степень, умножение, деление и деление по модулю выполняются раньше, чем сложение и вычитание.

Вычисленное значение переменной является ее стандартным числовым атрибутом.

Арифметические переменные. Арифметические переменные аналогичны арифметическим выражениям в алгоритмических языках. Переменная задается оператором **VARIABLE,** называемым оператором описания переменной, который содержит арифметическое выражение. Формат оператора описания переменной (табл. 15):

Табл. 15. Формат оператора

Поле	Информация, задаваемая в поле
Метка	Имя (числовое или символьное) переменной
Операция	**VARIABLE**
Операнд **A**	Выражение, которое используется для вычисления значения переменной

При любом обращении к переменной **RSL** (употребляется обозначение **V$RSL)** ее значение вычисляется как текущая длина очереди **WAITL (QTSWAITL** – СЧА регистратора очереди) плюс константа 3 и минус произведение значения функции **DSTRB** на значение параметра 7 транзакта, обрабатываемого в данный момент. В приведенном выражении **FN** – СЧА для обращения к функции, а **P** – СЧА транзакта.

Перед выполнением любой арифметической операции определяется значение каждого элемента и выделяется его целая часть. Постоянные без знака считаются положительными числами.

В выражении арифметической переменной могут быть использованы любые СЧА, функции и другие арифметические переменные. Запрещается использование самой вычисляемой переменной, а также переменных со знаком, так как знаки в данном случае рассматриваются как арифметические операции.

Система моделирования GPSS допускает использование скобок в выражениях арифметических переменных (для группировки членов или для обозначения операции умножения).

В GPSS World выражения, записанные в круглых скобках, обрабатываются вычислительной процедурой встроенного алгоритмического языка PLUS. Поэтому их можно использовать в качестве операндов блоков и операторов языка GPSS.

l. В GPSS/PC выражение может содержать не больше пяти пар скобок (не считая скобок, используемых при описании элементов матриц).

2. Пробелы между символами в выражениях не допускаются. Левый пробел записи считается концом выражения. Для записи выражения, превышающего длину строки, можно ввести другой оператор **VARIABLE** с именем, отличным от имени первой переменной, и включить значение новой переменной в качестве одного из операндов в выражение первой арифметической переменной.

Арифметические переменные с плавающей точкой аналогичны рассмотренным арифметическим переменным, за исключением того, что все операции над операндами выражений переменных с плавающей точкой выполняются без преобразования операндов и промежуточных результатов в целые значения. Лишь окончательный результат вычисления преобразуется в целое число.

Формат операторов описания арифметических переменных с плавающей точкой идентичен рассмотренному выше формату операндов описания арифметических переменных за исключением того, что в поле операции записывается слово **FVARIABLE.** Правила написания операторов те же, что и для арифметических переменных. Арифметическая переменная и переменная с плавающей точкой не могут иметь одинаковые номера. Если они имеют одинаковые номера, то при вычислении используется более позднее из двух описаний.

Различие результатов, полученных при вычислении с плавающей точкой и фиксированной, можно увидеть из такого примера:

```
FLOAT      FVARIABLE      10#(ll/3)
FIXED      VARIABLE       10#(11/3)
```

Значение переменной **FLOAT** равно 36, так как константа 10 умножается на 3,67 и от результата 36,7 взята целая часть. Переменная **FIXED** равна 30, так как результат

промежуточной операции деления будет округлен до 3.
1. Для переменных с плавающей точкой не допускается операция деления по модулю.
2. Использование дробных констант допускается только при описании переменных с плавающей точкой.
3. Стандартный числовой атрибут V**$<имя переменной>** используется для обращения к значениям как арифметических переменных, так и переменных с плавающей точкой. Способ вычисления переменной определяется оператором описания этой переменной.

Булевы переменные. Булевы переменные позволяют принимать решения в зависимости от значений СЧА и состояния объектов GPSS, используя для этого только одно выражение.

Булевы переменные – это логические выражения, состоящие из различных СЧА и (или) других булевых переменных. В булевой переменной проверяется одно или несколько логических условий. Результатом проверки есть единица (истина), если условия выполняются, и ноль (ложь) – в противном случае.

При описании булевых переменных используются три типа операторов: логические, булевы и операторы отношений.

Логические операторы связаны с такими ресурсами, как устройства, МКУ и логические ключи. Они используются для определения состояния данных объектов. Логические операторы, используемые в GPSS, представлены в табл. 16.

Табл. 16. Логические операторы

Логические операторы	Значение оператора, отражающее состояние ресурса
FVj или **Fj**	Равно 1, если устройство *j* занято или обслуживает прерывание, в противном случае – 0
FNVj	Равно 1, если устройство *j* не занято и не обслуживает прерывание, в противном случае – 0
Ij	Равно 1, если устройство *j* обслуживает прерывание, в противном случае – 0
NIj	Равно 1, если устройство *j* не обслуживает прерывание, иначе – 0
NUj	Равно 1, если устройство *j* не используется, в противном случае – 0
Uj	Равно 1, если устройство *j* используется, в противном случае – 0
SFj	Равно 1, если МКУ *j* заполнено, иначе – 0
SNFj	Равно 1, если МКУ *j* не заполнено, иначе – 0
SEj	Равно 1, если МКУ *j* пусто, иначе – 0
SNEj	Равно 1, если МКУ *j* не пусто, иначе – 0
SVj	Равно 1, если МКУ *j* находится в состоянии использования, в противном случае – 0
SNVj	Равно 1, если МКУ *j* не используется, в противном случае – 0
LRj	Равно 1, если логический ключ *j* выключен, иначе – 0
LSj	Равно 1, если логический ключ j включен, иначе – 0

Операторы отношения выполняют алгебраическое сравнение операндов. Операндами могут быть константы или стандартные числовые атрибуты. Все операторы отношений записываются в кавычках:

"G" (Greater) – больше;
"L" (Less) – меньше;
"E" (Equal) – равно;
"NE" (Not Equal) – не равно;
"LE" (Less than or Equal) – меньше или равно;
"GE"(Greater than or Equal) – больше или равно.

Есть два булевых оператора: "**OR**" – оператор «**ИЛИ**», и "**AND**" – оператор «**И**». Оператор «**ИЛИ**» проверяет, выполняется ли хотя бы одно из проверяемых условий. Оператор «**И**» требует выполнения обоих условий.

4.11. Определение функции в GPSS

В GPSS рассматриваются пять типов функций:

1) дискретная числовая (**D**),
2) непрерывная числовая (**C**),
3) табличная числовая (**L**),
4) дискретная атрибутивная (**E**),
5) табличная атрибутивная (**M**).

Рассмотрим два первых типа функций.

Дискретная функция представляет собой кусочно-постоянную функцию, которая состоит из горизонтальных ступеней (рис. 5). Непрерывная функция представляет собой кусочно-непрерывную функцию. Непрерывная функция в GPSS состоит из соединенных между собой прямых отрезков и представляет собой ломаную линию (рис. 6). Чтобы задать дискретную функцию, необходимо задать координаты крайних правых точек горизонтальных отрезков. Для непрерывной функции необходимо задать координаты всех точек, которые являются концами отрезков.

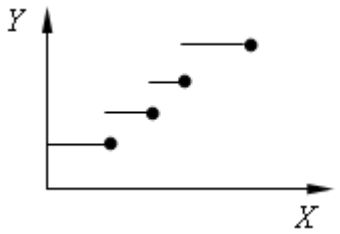


Рис. 5. Кусочно-постоянная функция

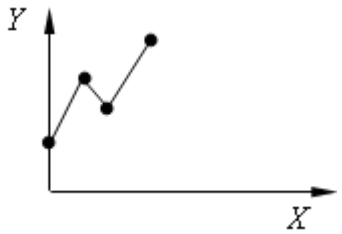


Рис. 6. Непрерывная функция в GPSS

Действия, необходимые для определения дискретной и непрерывной GPSS-функции:

1. Присвоить функции имя. Имя может быть числовым либо символьным.

2. Задать аргумент функции. Аргументом могут быть:

1) ссылка на генератор случайных чисел, используемый для розыгрыша в соответствии с распределением, заданным функцией;

2) стандартный числовой атрибут;

3) ссылка на любую другую функцию.

В первом случае аргумент задается в виде **RNj,** *j* – целое число (номер генератора). В GPSS/PC; = 1,..., 7, т.е. возможно обращение к семи идентичным генераторам случайных чисел. При этом генераторы выдают случайные числа в диапазоне 0... 0,999. В GPSS World количество генераторов случайных чисел неограниченно, а выдаваемые ими значения 0... 0,999999.

3. Задать тип функции и число крайних точек функции.

4. Задать значения аргумента (переменной) и соответствующие значения функции (т.е. координаты крайних точек функции).

Три первых элемента информации указываются в операторе определения функции. Формат оператора представлен в табл. 17.

Табл. 17. Формат оператора

Поле	Информация, задаваемая в поле
Операция	**FUNCTION**
Операнды **A** **B**	 **RNj** (j = 1, …, 7) или СЧА **Dn** либо **Cn**, где **D** определяет дискретную функцию, **C** определяет непрерывную функцию, n – для дискретной функции – это число различных значений, получаемых функцией (количество горизонтальных отрезков), для непрерывной функции – это число, на единицу больше числа отрезков, составляющих функцию (количество точек)

За каждым оператором описания **FUNCTION** следуют операторы задания координат точек функции (значений аргументов X_i и соответствующих им значений функции Y_i,) – это ***операторы описания координат функции.*** Их формат:

1) если координаты всех точек расположены в одной строке оператора описания функции:

$$X_1,\ Y_1 / X_2,\ Y_2 / \ldots / X_n,\ Y_n,$$

2) если координаты точек расположены в нескольких операторах описания функции:

$$X_1,\ Y_1 / X_2,\ Y_2 / \ldots / X_i,\ Y_i,$$

$$X_{i+1},\ Y_{i+1} / X_{i+2},\ Y_{i+2} / \ldots / X_k,\ Y_k,$$

$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots,$$

$$X_{m+1},\ Y_{m+1} / X_{m+2},\ Y_{m+2} / \ldots / X_n,\ Y_n,$$

где X_i и Y_i – координаты i-й точки функции (в случае моделирования случайной величины X_i является i-й суммарной (кумулятивной) частотой, Y_i – соответствующим значением случайной величины).

Особенности оператора описания координат функции:

1) основной единицей информации оператора описания координат функции является пара значений X_i, Y_i (координаты точки i);

2) значения координат X_i и Y_i одной точки функции разделяются запятой;

3) последовательные наборы координат разделяются знаком «/»;

4) координаты X_i и Y_i относящиеся к одной точке, задаются одним оператором, т.е. пара координат одной точки не должна разрываться;

5) все строки описания координат функции должны начинаться с первой позиции;

6) во всех случаях значения аргумента должны удовлетворять следующим неравенствам:

$$X_1 < X_2 < \ldots < X_i < \ldots < X_n.$$

Значение функции является ее стандартным числовым атрибутом. Способ ссылки на этот атрибут зависит от того, как задано имя функции: в символьном или числовом виде. Если имя числовое, то к значению функции обращаемся через **FNj** (где j – номер функции), если имя символьное – через **FN$<имя функции>**.

1. Аргументом функции может быть и значение какой-либо другой функции.
2. Каждая функция должна иметь, по крайней мере, две описанные точки.

Моделирование неравномерных случайных величин. Использование функций в блоках GENERATE и ADVANCE. Пусть распределение интервалов поступления через определенный блок **GENERATE** или время задержки в некотором блоке **ADVANCE** не является равномерным (либо является равномерным с «плавающими во времени», т.е. нефиксированными значениями среднего и половины поля допуска). Для входов транзактов в модель через этот блок **GENERATE** и для задания закона времени задержки в соответствующем блоке **ADVANCE** необходимо использовать функции и (или) СЧА. Использование функций, заданных в операндах блоков, зависит от контекста. От значения функции берется целая часть, за исключением тех случаев, когда это значение используется в качестве операнда **B** блоков **GENERATE и ADVANCE** или операнда C блока **ASSIGN.** В табл. 18 показаны различные варианты использования функций и СЧА в качестве операндов **A** и **B** блоков **GENERATE и ADVANCE.** Под результатом понимается значение интервала поступления или задержки.

Табл. 18. Варианты использования функций и СЧА

Операнд **A**	Операнд **B**	Результат
α (число или СЧА)	β (число или СЧА)	Генерируется случайное число, равномерно распределенное на интервале $\alpha \pm \beta$. Результат равен полученному числу
FN$DIS	Отсутствует	Результат равен значению функции **DIS**
Отсутствует	**FN$B**	Данная комбинация не доступна
FN$DIS	β (число или СЧА)	Вначале вычисляется значение функции **DIS**. Берется целая часть этого значения (пусть это будет число α), после чего генерируется случайное число, равномерно распределенное на интервале $\alpha \pm \beta$. Результат равен полученному числу.
α (число или СЧА)	**FN$DIS**	Вначале вычисляется значение функции **DIS** (пусть это будет число β). После чего находится произведение $\alpha \times \beta$. Результат равен целой части этого произведения.
FN$DIS1	**FN$DIS2**	Вычисляется значение функции DIS1 и **DIS2** (пусть это будут числа α и β). После чего находится произведение $\alpha \times \beta$. Результат равен целой части этого произведения.

Непрерывные случайные переменные, рассматриваемые как дискретные. Как известно, дискретные случайные переменные могут принимать только фиксированное число значений. В противоположность этому, непрерывные (в классическом смысле этого термина) случайные переменные могут иметь неограниченное число различных значений.

На практике обычно достаточно, чтобы все случайные переменные имели конечное число конкретных значений. Нет необходимости в тщательном определении значений этих случайных переменных, за исключением случаев, когда необходимо делать расчеты с высокой степенью точности. Таким образом, вполне возможна дискретизация непрерывных распределений. После этого они могут быть определены в GPSS с помощью дискретных и непрерывных GPSS-функций (непрерывные GPSS-функции по сути также являются дискретными, поскольку множество их значений дискретно и конечно).

Функции распределения случайных величин. В языке GPSS возможность задания функций распределения случайных величин ограничена заданием их в табличном

виде путем аппроксимации непрерывными функциями. Поэтому можно задать только те функции, которые легко преобразовать для новых значений параметров. К таким функциям, например, относится функция экспоненциального распределения с параметром λ, = 1, а также функция стандартного нормального распределения с математическим ожиданием $m = 0$ и стандартным отклонением σ =1.

Эти ограничения не касаются языка GPSS World, в котором для задания различных вероятностных функций распределения можно использовать библиотечные процедуры, написанные на языке PLUS. Однако использование вероятностных распределений в табличном виде значительно ускоряет процесс моделирования.

Моделирование пуассоновского потока. Рассмотрим табличный способ задания пуассоновского потока заявок. Пуассоновский входящий поток описывается таким образом: вероятность поступления k заявок пуассоновского потока в течение интервала t составляет

$$p_k(t) = \frac{e^{-ët}(ët)^k}{k!},\ k = 1, 2, \ldots,$$

где λ – интенсивность потока.

Интервалы времени между соседними заявками пуассоновского потока распределены по экспоненциальному закону. Согласно методу обратной функции, можно получить ряд чисел, которые имеют экспоненциальное распределение, если ряд случайных чисел R, равномерно распределенных на интервале [0,1], преобразовать в соответствии с функцией, обратной к экспоненциальной функции распределения:

$$t_j = F^{-1}(x) = -\overline{T}\ln(r_j),$$

где t_j - 7-й разыгранный интервал времени поступления; T = 1/λ, -средний интервал времени поступления; r_j – j-е число в последовательности случайных чисел R с равномерным распределением на интервале [0, 1].

Разработчиками GPSS была осуществлена аппроксимация функции $F^{-1}(x)$, обратной к экспоненциальной функции распределения с параметром λ = 1. Таким образом, функция $F^{-1}(x)$ была заменена 23 отрезками, которые использовались для преобразования значений **RNj** в значение – **In(RNj).**

Функция **XPDIS** определяет экспоненциальное распределение с интенсивностью λ = 1.

Пуассоновский входящий поток с интенсивностью λ, отличной от единицы, моделируется с помощью блока **GENERATE** таким образом:

1) в качестве операнда **A** используют среднее значение интервалов времени T= λ/k, где λ – интенсивность пуассоновского потока;

2) в качестве операнда **B** используют СЧА – значение функции **XPDIS,** операторы определения и описания которой приведены выше.

Моделирование гипер- и гипоэкспоненциального распределений. Экспоненциальную функцию распределения можно использовать также для моделирования гипер- и гипоэкспоненциального распределений.

Неэкспоненциальное распределение с коэффициентом вариации $C > 1$ (C – это отношение стандартного отклонения к математическому ожиданию случайной величины) можно получить с помощью взвешенной суммы экспонент – гиперэкспоненциального распределения:

$$F_i(x) = P(t < x) = \sum_{i=1}^{k} \omega_i \left(1 - e^{-\mu_i x}\right), \quad \mu_i > 0, \quad \omega_i > 0, \quad \sum_{i=1}^{k} \omega_i = 1;$$

$$\bar{t} = \sum_{i=1}^{k} \frac{\omega_i}{\mu_i}; \quad D(t) = \sum_{i=1}^{k} \frac{2\omega_i}{\mu_i^2} - \left(\sum_{i=1}^{k} \frac{\omega_i}{\mu_i}\right)^2; \quad C = \frac{\sqrt{D(t)}}{\bar{t}} \geq 1$$

Если $\mu_i = \mu$ для всех i, то $C = 1$ – имеем экспоненциальное распределение.

Гиперэкспоненциальное распределение можно получить при параллельном соединении k (рис. 7) экспоненциальных обслуживающих устройств с интенсивностью обслуживания μ_i и вероятностью ω_i использования для обслуживания (i = 1, k). Причем в произвольный момент времени может быть занято не более одного устройства из k. Такое распределение хорошо описывает распределение времени работы центрального процессора компьютера.

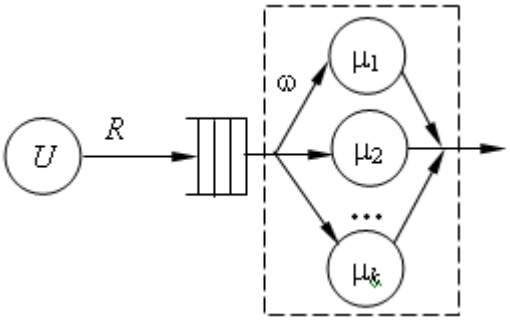


Рис. 7. Параллельное соединение

Для моделирования гиперэкспоненциального распределения со средним значением 6,28 и стандартным отклонением 8,4 необходимо определить переменную

HYP FVARIABLE (410+(RN2 'L' 234)(#(1334 - 410)))#FN$XPDIS

Эту переменную можно использовать в блоке задержки так:

ADVANCE VSHYP

Гипоэкспоненциолъное распределение с коэффициентом вариации $C < 1$ описывается таким образом:

$$F_i(x) = P(t \leq x) = 1 - \frac{\mu_2}{\mu_2 - \mu_1} e^{-\mu_1 x} - \frac{\mu_1}{\mu_1 - \mu_2} e^{-\mu_2 x}, \ (\mu_1 \neq \mu_2);$$

$$\bar{t} = \frac{1}{\mu_1} + \frac{1}{\mu_2}; \quad C = \sqrt{1 - \frac{2\mu_1\mu_2}{(\mu_1 + \mu_2)^2}} < 1; \quad D(t) = \frac{1}{\mu_1^2 + \mu_2^2}.$$

При равенстве всех коэффициентов μ распределение времени пребывания в обслуживающем центре (на рис. 7 обведен пунктирной линией) будет *k-распределением* Эрланга:

$$F_i(x) = 1 - e^{-k\mu x} \sum_{i=0}^{k-i} \frac{(k\mu x)^2}{i!}$$

Гипоэкспоненциальное распределение характерно, например, для времени обслуживания устройств ввода–вывода. Его можно получить последовательным соединением обслуживающих экспоненциальных устройств, причем в любой момент времени должно быть занято не более одного устройства (рис. 8).

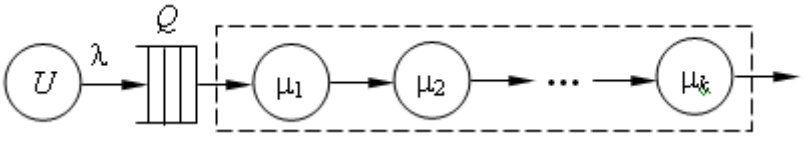


Рис. 8. Гипоэкспоненциальное распределение

Моделирование эрланговского потока. Экспоненциальное распределение не всегда адекватно описывает время обслуживания и поступления требований в систему. Более реалистичным является распределение Эрланга. В то же время, это распределение является частным случаем гамма-распределения, которое описано ниже. Для потока Эрланга k-го порядка с интенсивностью λ математическое ожидание и дисперсия определяются так: $\mathrm{E}[x] = 1/(k\lambda)$, $\mathrm{D}[x] = 1/(k\lambda^2)$. Для моделирования распределения Эрланга может также использоваться экспоненциальная функция распределения, для этого достаточно просуммировать *k* случайных экспоненциально распределенных величин. С ростом *k* распределение Эрланга будет приближаться к нормальному распределению. Например, поток Эрланга второго порядка со средним значением времени поступления 180 можно задать таким образом:

```
GENERATE    ,,1
SDFG  ADVANCE      90,FN$EXPDIS
ADVANCE      90,FN$EXPDIS
SPLIT  1,SDFG
TERMINATE   1
```

В нулевой момент времени в модель вводится транзакт. Этот транзакт в каждом их двух последующих блоков **ADVANCE** задерживается на экспоненциально распределенный промежуток времени. Блок **SPLIT** создает копию транзакта и направляет ее на блок с меткой **SDFG,** исходный транзакт поступает в модель и т.д.

Моделирование нормального закона распределения. Функция стандартного нормального закона распределения с параметрами $m = 0$, $a = 1$ задается в GPSS 24 отрезками следующим образом:

```
NOR    FUNCTION      RN1,C2S
0,-5/.00003,-4/.00135,-3/.00621,-2.5/.02275,-2
.06681,-1.5/.11507,-1.2/.15866,-1/.2П86,-.8/.27425,-.6
.34458,-.4/.42074,-.2/.5,0/.57926,.2/.65542,.4
.72575,.6/.78814,.8/.84134,1/.88493,1.2/.93319,1.5
.97725,2/.99379,2.5/.99865,3/.99997,4/1,5
```

Для того чтобы получить функцию нормального распределения случайной величины *X* с математическим ожиданием $m_x \neq 0$ и среднеквадратичным отклонением $\sigma_x \neq 1$, необходимо произвести вычисления по формуле:

$$X = m_x + \sigma_x Z,$$

где Z – случайная величина со стандартной нормальной функцией распределения. Например, если случайная величина *X* имеет параметры $m_x = 60$ и $\sigma_x = 10$, то в GPSS эта случайная величина моделируется так:

NOR1 FVARIABLE 60+10#FN$NOR

Если необходимо осуществить задержку по этому закону распределения, то используется блок

ADVANCE VSNOR1

При использовании функции нормального распределения для блоков **GENERATE и ADVANCE** необходимо обеспечить неотрицательность значений интервалов поступления и задержки. Это можно сделать, если $m_x \geq 5\sigma_x$.

Моделирование вероятностных функций распределения в GPSS World. В GPSS World в библиотеку процедур включено 24 вероятностных распределений. При вызове вероятностного распределения требуется определить аргумент ***Stream*** (может

быть выражением), который определяет номер генератора случайных чисел. При моделировании генераторы случайных чисел создаются по мере необходимости и их явное определение не обязательно. Большинство вероятностных распределений имеют некоторые параметры. Аргументы процедур, называемые обычно ***Locate, Scale* и *Shape,*** часто используются для этих целей. Аргумент ***Locate*** используется после построения применяемого распределения и прибавляется к нему. Это позволяет горизонтально перемещать функцию распределения по оси X. Аргумент ***Scale*** обычно меняет масштаб функции распределения, а ***Shape*** – ее форму.

Встроенная библиотека процедур содержит следующие вероятностные распределения:

1) бета (Beta);
2) биномиальное (Binomial);
3) Вейбулла (Weibull);
4) дискретно-равномерное (Discrete Uniform);
5) гамма (Gamma);
6) геометрическое (Geometric);
7) Лапласа (Laplace);
8) логистическое (Logistic);
9) логлапласово (LogLaplace);
10) логлогистическое (LogLogistic);
11) логнормальное (LogNormal);
12) нормальное (Normal);
13) обратное Вейбулла (Inverse Weibull);
14) обратное Гаусса (Inverse Gaussian);
15) отрицательное биномиальное (Negative Binomial);
16) Парето (Pareto);
17) Пирсона типа V (Pearson Type V);
18) Пирсона типа VI (Pearson Type VI);
19) Пуассона (Poisson);
20) равномерное (Uniform);
21) треугольное (Triangular);
22) экспоненциальное (Exponential);
23) экстремального значения A (Extreme Value A);
24) экстремального значения B (Extreme Value B).

В качестве примера покажем, как для генерации потока транзактов можно использовать библиотечную процедуру экспоненциального распределения с параметром X = 0,25 и использованием генератора случайных чисел **RN1**:

GENERATE (Exponential(1,0,(1/0.25)))

Из всех приведенных распределений опишем те, которые наиболее часто используются на практике.

Логарифмически нормальное распределение. Логарифмически нормальное распределение (логнормальное) – это распределение случайной величины, натуральный логарифм которой нормально распределен. Это распределение пригодно для моделирования мультипликативных процессов так же, как нормальное – для аддитивных.

С помощью центральной предельной теоремы можно показать, что произведение независимых положительных случайных величин стремится к логарифмически нормальной случайной величине.

Логнормальная случайная величина формируется под влиянием большого числа независимых факторов, причем каждый отдельный фактор оказывает равномерно не-

значительное и равновероятное по знаку влияние. Прирост каждого следующего фактора пропорционален уже достигнутому к этому времени значению исследуемой величины. То есть рассмотренный характер воздействия является мультипликативным.

Функция плотности логнормального распределения:

$$f_\eta(x) = \frac{1}{\sqrt{2\pi}\sigma(x-\lambda)} e^{\frac{(\ln(x-\lambda)-\mu)^2}{2\sigma^2}},$$

если $x > \lambda$, в противном случае – $f_\eta(x) = 0$.

Если после логарифмирования каждого элемента некоторого набора данных этот трансформированный набор данных нормально распределен, то исходные данные логарифмически нормально распределены.

Это распределение используется при моделировании экономических, информационных, физических и биологических систем. Оно хорошо моделирует процессы в случае, когда значение наблюдаемой переменной является случайной долей от значения предыдущего наблюдения.

Примерами использования этого распределения могут быть:

1) размеры и вес частиц, образуемых при дроблении;
2) доход семьи;
3) зарплата работников;
4) долговечность изделия, работающего в режиме износа и старения;
5) размер банковского вклада;
6) длины слов в языке;
7) длины передаваемых сообщений.

Например, когда неизвестно распределение длины передаваемых сообщений, размера файлов или длины запроса к базе данных, то с большой вероятностью можно предположить логнормальное распределение для этих величин.

Математическое ожидание и дисперсия логнормально распределенной случайной величины таковы:

$$E_\eta = e^{\mu+\frac{\sigma^2}{2}} + \lambda, \quad D_\eta = e^{2\mu+\sigma^2}\left(e^{\sigma^2}+1\right),$$

где параметр σ задает среднеквадратическое отклонение, μ – математическое ожидание из нормального распределения, λ – величину сдвига для определения местоположения распределения.

Для вызова логнормального распределения используется библиотечная процедура

LOGNORMAL(Stream, Locate, Scale, Shape),

где ***Stream*** – номер генератора случайных чисел, автоматически преобразуется в целое число, которое должно быть больше или равно 1; ***Located* = λ, *Scale* = σ; *Shape* = μ.** Все параметры обязательные.

Гамма-распределение является обобщенным распределением Эрланга для случая, когда число a суммируемых величин – является нецелым. Гамма-распределенная величина имеет значения от 0 до ∞, то есть неотрицательна. Если α – целое, то это будет распределение Эрланга.

Функция распределения значительно изменяет свою форму при различных параметрах, что позволяет использовать это распределение для моделирования различных физических явлений.

Гамма-распределение можно интерпретировать как сумму квадратов нормально

распределенных случайных величин, то есть как χ^2-распределение.

Таким образом, χ2-распределение, распределение Эрланга и экспоненциальное распределение являются частными случаями гамма-распределения.

Математическое ожидание и дисперсия гамма-распределенной случайной величины таковы:

$$E_\gamma = \alpha\beta + \lambda, \quad D_\gamma = \alpha\beta^2.$$

Функция плотности гамма-распределения имеет вид:

$$f_\gamma(x) = \frac{\beta^{-\alpha}(x-\lambda)^{(\alpha-1)} e^{\frac{(x-\lambda)}{\beta}}}{\Gamma(\alpha)}, \quad x > \lambda,$$

где $0 \le x < \infty$; $f_\gamma(x) = 0$, если $x < 0$; $\Gamma(\alpha) = \int_0^\infty x^{\alpha-1} e^{-x} dx$ - гамма-функция Эйлера; параметр α задает форму распределения, β – масштаб для сжатия или растяжения распределения, λ – величину сдвига для определения местоположения распределения.

Для вызова гамма-распределения используется библиотечная процедура

GAMMA (Stream, Locate, Scale, Shape),

где ***Stream*** – номер генератора случайных чисел, автоматически преобразуется в целое число, которое должно быть больше или равно 1; ***Located* = λ, *Scale* = β; *Shape* = α.** Все параметры обязательные.

Когда аргумент ***Shape*** равен 1, гамма-распределение вырождается в экспоненциальное. Это означает, что **GAMMA (Stream, Locate, Scale, 1)** имеет то же распределение, что и **EXPONENTIAL (Stream, Locate, Scale).**

Распределение Вейбулла. Это распределение используется при моделировании жизненного цикла сложного изделия или индивидуума.

Функция плотности распределения Вейбулла имеет вид:

$$f(x) = \frac{\alpha(x-\alpha)^{\alpha-1}}{\beta^\alpha} e^{-\left(\frac{x-\alpha}{\beta}\right)^\alpha}$$

При $x > \alpha$, в противном случае – $f(x) = 0$.

Математическое ожидание и дисперсия:

$$E = \frac{\beta}{\alpha}\Gamma\left(\frac{1}{\alpha}\right) + \lambda, \quad D = \frac{\beta^2}{\alpha}\left(2\Gamma\left(\frac{2}{\alpha}\right) - \frac{1}{\alpha}\left(\Gamma\left(\frac{1}{\alpha}\right)\right)^2\right)$$

где $\Gamma(\alpha)$ – гамма-функция Эйлера, параметр α задает форму распределения, β – интенсивность отказов, λ – величину сдвига для определения местоположения распределения.

Для вызова распределения Вейбулла используется библиотечная процедура

WEIBULL (Stream, Locate, Scale, Shape),

где *Stream* номер генератора случайных чисел, автоматически преобразуется в целое число, которое должно быть больше или равно 1; ***Located* = λ, *Scale* = β; *Shape* = α.** Все параметры обязательные.

Когда аргумент ***Shape*** равен 1, распределение Вейбулла вырождается в экспоненциальное. Это означает, что **WEIBULL (Stream, Locate, Scale, 1)** имеет то же распреде-

ление, что и **EXPONENTIAL (Stream, Locate, Scale).**

4.12. Стандартные числовые атрибуты, параметры транзактов. Блоки ASSIGN, MARK, LOOP

В процессе моделирования интерпретатор автоматически регистрирует и корректирует информацию, касающуюся различных элементов, используемых в модели. Большая часть информации доступна только интерпретатору и используется для сбора статистической информации о работе модели. Однако к некоторым атрибутам (свойствам) объектов может обращаться и программист, управляя процессом моделирования в зависимости от их значений.

Рассмотрим несколько примеров зависимости функционирования элементов модели от системных атрибутов, т.е. СЧА.

1. Интенсивность работы некоторого устройства зависит от длины очереди. Для определения времени обслуживания при каждом поступлении транзакта на обслуживание необходимо знать значение такого системного атрибута, как длина очереди.

2. Интенсивность обслуживания некоторого устройства зависит от общей продолжительности его функционирования (проявление усталости - интенсивность со временем уменьшается, разогрев устройства – интенсивность со временем увеличивается). Время обслуживания – функция, которая зависит от времени, прошедшего с начала работы.

3. Имеются два устройства и диспетчер, который распределяет работы между ними таким образом, чтобы загрузка устройств была равномерной. Для этого в пункте диспетчеризации необходимо иметь информацию о коэффициентах загрузки устройств и возможность выбора пути перемещения транзакта в зависимости от этих двух величин.

Условно атрибуты можно поделить на две категории:

1) атрибуты системы;

2) атрибуты транзактов.

Атрибуты системы - это параметры, которые описывают состояние объектов модели. Такие количественные показатели, как «текущая длина очереди» или «коэффициент загрузки устройства» являются типичными системными атрибутами. Стандартный набор атрибутов, подобных указанным, автоматически поддерживается интерпретатором GPSS.

Транзакты также могут иметь некоторые числовые характеристики (например, уровень приоритета). Кроме того, транзакт снабжается некоторым числом параметров.

В языке GPSS атрибуты (свойства) объектов – это СЧА. Каждый объект GPSS имеет свой набор СЧА. Доступ к СЧА осуществляется при использовании специальных обозначений этих атрибутов. Имя СЧА состоит из двух частей:

1) групповое имя – состоит из одной или двух букв, идентифицирует тип объекта и тип информации о нем;

2) имя конкретного члена группы.

Объекты могут идентифицироваться с помощью числовых и символьных имен. Если объект идентифицируется с помощью номера (числовое имя), то ссылка на его стандартный числовой атрибут записывается как **C4Aj**, где *j* – номер объекта (целое число). При символьной идентификации объекта ссылка на его стандартный числовой атрибут записывается, как **СЧА$<имя объекта>** (в приведенных ссылках под «СЧА» понимается групповое имя).

Стандартные числовые атрибуты. В табл. 19 показаны СЧА устройств, в табл. 20 – СЧА МКУ, в табл. 21 – СЧА очередей.

Табл. 19. СЧА устройств

Обозначение	Значение
Fj или **F$имя**	Показатель занятости устройства (0 – если не занято, 1 – если занято)
FCj или **FC$имя**	Число занятий устройства
FRj или **FR$имя**	Нагрузка устройства, выраженная в долях тысячи
FTj или **FT$имя**	Целая часть значения среднего времени задержки транзакта в устройстве
FVj или **FV$имя**	Флаг готовности устройства к использованию (1 – готово, 0 – в противном случает)

Табл. 20. СЧА МКУ

Обозначение	Значение
Rj или **R$имя**	Емкость незаполненной части МКУ
Sj или **S$имя**	Емкость заполненной части МКУ
SAj или **SA$имя**	Целая часть среднего заполнения МКУ
SCj или **SC$имя**	Счетчик числа входов в МКУ. (При каждом выполнении блока **ENTER** значение счетчика увеличивается на значение операнда B этого блока)
SMj или **SM$имя**	Максимально занятая емкость МКУ. Запоминает максимальное значение **Sj** (**S$имя**)
SRj или **SR$имя**	Нагрузка МКУ, выраженная в долях тысячи
STj или **ST$имя**	Целая часть среднего времени пребывания транзакта в МКУ
SVj или **SV$имя**	Флаг готовности МКУj к использованию (1 – готово, 0 – не готово)

Табл. 21. СЧА очередей

Обозначение	Значение
Qj или **Q$имя**	Текущее значение длины очереди (текущее содержимое)
QAj или **QA$имя**	Целая часть среднего значения длины очереди
QCj или **QC$имя**	Число входов в очередь. При каждом входе в блок **QUEUE** очереди значение **QCj** (**QC$имя**) увеличивается на значение операнда **B**, при каждом входе в блок **DEPART** очереди значение **QCj** (**QC$имя**) уменьшается на значение операнда **B**
QMj или QM$имя	Максимальное значение длины очереди (максимальное значение **Qj** (**Q$имя**))
QTj или **QT$имя**	Целая часть среднего времени пребывания в очереди всех транзактов, которые входили в очередь (включая и те, которые не ждали – нулевые входы)
QXj или **QX$имя**	Целая часть среднего времени пребывания в очереди для транзактов, которые ждали в очереди (ненулевые входы)
QZj или **QZ$имя**	Число нулевых входов в очередь

СЧА блоков и системные СЧА. Блоки имеют два стандартных числовых атрибута (их подсчет ведется автоматически):

Wj (W$<метка блока>) – счетчик текущего содержимого блока с номером *j* (с меткой блока);

Nj (N$<метка блока>) – счетчик входов, т.е. общее число транзактов, вошедших в блоку с момента последнего действия операторов **RESET** и **CLEAR** (от начала работы модели, если не было операторов **RESET** и **CLEAR).**

Важные системные СЧА:

C1 – текущее значение относительного модельного времени; автоматически изме-

няется интерпретатором и устанавливается в 0 при выполнении операторов **CLEAR** и **RESET;**

AC1 – текущее значение абсолютного модельного времени; автоматически изменяется интерпретатором и устанавливается в 0 при выполнении оператора **CLEAR;**

TG1 – текущее значение счетчика завершения;

PR – приоритет транзакта, обрабатываемого в данный момент;

Ml – время пребывания в модели транзакта, обрабатываемого интерпретатором в данный момент.

Параметры транзактов. Параметры транзактов – это свойства транзакта, определяемые пользователем. Множество параметров транзакта – набор стандартных числовых атрибутов, которые принадлежат транзакту. Параметры транзакта являются локальными переменными, которые доступны только данному транзакту.

В процессе перемещения транзакта по модели, его параметры (могут задаваться и модифицироваться в соответствии с логикой работы модели.

Особенности параметров транзактов:

1. Доступ к параметрам транзактов осуществляется таким образом:

P<номер> или P$<имя>,

где **P** – СЧА транзакта, определяющий его групповое имя, т.е. имя всех параметров транзакта.

2. Номера (имена) конкретных членов множества параметров задаются с помощью целых чисел 1, 2, ... или символьных имен. Например, **P22** – это 22-й параметр транзакта, **P$COLOR** – параметр с именем **COLOR.**

3. При входе транзакта в модель начальное значение всех его параметров устанавливается в ноль.

4. Значения параметров транзактов и их изменение определяет пользователь.

5. Значениями параметров транзактов могут быть любые числа (в системе GPSS/PC – только целые числа). Параметры могут приобретать отрицательные значения.

6. Транзакт может обращаться только к своим параметрам. Если необходимо получить доступ к параметрам других транзактов, то это можно сделать только через ячейки сохраняемых величин или используя группы транзактов.

7. Параметры можно использовать в качестве операндов блоков и в качестве аргументов функций.

8. Параметры позволяют организовать косвенную адресацию блоков. Это дает возможность агрегированного представления объектов моделирования в программе.

Изменение значений параметров. Блок ASSIGN НАЗНАЧИТЬ). При входе транзакта в этот блок значения параметров могут задаваться или изменяться. Формат блока (значение операндов приведено в табл.22):

ASSIGN A[+,–],B[,C]

Табл.22. Значение операндов

Операнд	Значение	Результат по умолчанию
A	Номер или имя модифицируемого или задаваемого параметра	Ошибка
B	Величина, используемая для модификации (число или СЧА)	Ошибка
C	Имя функции	Не используется

Блок **ASSIGN** может быть использован как в режиме замещения значения пара-

метра (начальное значение всех параметров транзактов равно 0), так и в режиме увеличения и уменьшения. В режиме увеличения предшествующее значение параметра увеличивается на значение, стоящее в операнде **B**. В режиме уменьшения оно уменьшается на величину, стоящую в операнде **B**. Режимы увеличения и уменьшения определяются введением соответственно знаков «плюс» и «минус» перед запятой, которая разделяет операнды **A** и **B**.

При использовании операнда **C** значение операнда **B** умножается на значение функции, указанной в операнде **C**. Параметр, заданный в операнде **A**, изменяется на величину полученного произведения (в режиме увеличения и уменьшения) или приобретает значение результата (в режиме замещения).

Отметка времени. При каждом входе транзакта в модель интерпретатор фиксирует для него текущее значение времени. Это значение времени называется отметкой времени. Она может быть интерпретирована как время «рождения» транзакта или время входа транзакта в модель. В явном виде отметка времени недоступна. Однако существует СЧА, который тесно связан со значением времени входа транзакта в модель. Его имя **Ml**, а значение определяется так:

$$M1 = \begin{pmatrix} \text{Текущеез значение} \\ \text{таймера абсолютного} \\ \text{времени} \end{pmatrix} - \begin{pmatrix} \text{Значение времени} \\ \text{входа транзакта} \\ \text{в модель} \end{pmatrix}$$

Значение **M1** для каждого транзакта изменяется в процессе моделирования. Сразу после входа транзакта в модель **M1** = 0, через 10 единиц модельного времени **M1** = 10 и т.д.

Транзитное время. Блок MARK (ОТМЕТИТЬ). Стандартный числовой атрибут **M1** измеряет время, которое прошло с момента входа транзакта в модель. Однако очень часто требуется знать время, затраченное на перемещение транзакта между двумя произвольными точками модели. Для этого используется блок **MARK.**

При входе транзакта в блок **MARK** значение таймера абсолютного времени записывается в качестве одного из его параметров. Такую запись называют отметкой транзакта. Формат блока **MARK (**значение операнда приведено в табл.23**):**

MARK A

Табл. 23. Значение операнда

Операнд	Значение	Результат по умолчанию
A	Номер параметра, в который записывается значение абсолютного времени (целое число, СЧА)	При отсутствии операнда А отметка времени заменяется текущим значением абсолютного времени

Пусть необходимо определить интервал времени, на протяжении которого транзакт проходит от точки T_1 к точке T_2. Для этого нужно выполнить два действия:

1) в точку T_1 поместить блок **MARK j,** где j – номер параметра, в который записывается значение абсолютного времени в момент записи;

2) в точке T_2 обратиться к СЧА с именем **MPj**, где *j* – номер параметра, в котором сделана отметка времени транзакта; СЧА **MPj** будет иметь такое значение:

$$MPj = \begin{pmatrix} \text{Текущеез значение} \\ \text{таймера абсолютного} \\ \text{времени} \end{pmatrix} - \begin{pmatrix} \text{Значение} \\ \text{j - го} \\ \text{параметра} \end{pmatrix}$$

Организация циклов. Блок LOOP (ЦИКЛ). С помощью параметров транзактов в программе можно организовать циклы. Для этого используется блок **LOOP.** Он

управляет количеством повторных прохождений транзактом определенной последовательности блоков модели.

Формат блока (значение операндов приведено в табл. 24):

LOOP A[,B]

Табл. 24. Значение операндов

Операнд	Значение	Результат по умолчанию
A	Параметр транзакта, используемый для организации цикла (переменная цикла). Он может быть именем, положительным целым числом, СЧА, СЧА*СЧА (косвенная адресация)	Ошибка
B	Метка (имя блока) начального блока цикла	Ошибка

Когда транзакт входит в блок **LOOP,** параметр, указанный в операнде **A**, уменьшается на единицу, а затем проверяется его значение на равенство нулю. Если значение не равно нулю, то транзакт переходит в блок, указанный в операнде **B**. Если значение параметра равно нулю, транзакт переходит в следующий блок.

Переменная блока **LOOP** может только уменьшаться.

4.13. Изменение приоритета транзактов. Блок PRIORITY

Блок **RIORITY (НАЗНАЧИТЬ ПРИОРИТЕТ)** присваивает или изменяет приоритет транзакта, если он был задан блоком **GENERATE** (по умолчанию приоритет транзакта равен нулю). Его формат (значение операндов приведено в табл. 25):

PRIORITY A[,B]

Табл. 25. Значение операндов

Операнд	Значение	Результат по умолчанию
A	Новое значение приоритета (целое число, СЧА, СЧА*СЧА)	Ошибка
B	Этот операнд определяет режим **BUFFER**	

Новое значение приоритета может быть меньше, больше или равно текущему значению приоритета транзакта. Приоритет влияет на порядок выбора транзакта для обслуживания устройствами и на порядок просмотра транзактов в списке текущих событий [7].

Стандартный числовой атрибут этого блока – **PR**. Поскольку уровень приоритета транзакта может изменяться от 0 до 127, то **PR** будет выдавать значение в диапазоне 0–127.

4.14. Организация обслуживания с прерыванием. Блоки PREEMPT и RETURN

Во многих случаях возникает необходимость организации обслуживания в устройстве с прерываниями (например, при выполнении некоторой операции на станке произошла его поломка). Такую ситуацию можно смоделировать, считая, что отказ оборудования представляет собой транзакт, приоритет которого выше, чем у транзакта, обрабатываемого станком. В этом случае более приоритетный транзакт должен прервать обслуживание менее приоритетного транзакта, т.е. выгрузить его из устройства. Отсюда понятен дословный перевод с английского слова ***preempt*** – выгрузить, но с точки зрения работы одноканальной СМО принято использовать термин **ЗАХВАТИТЬ** устройство

[10]. Для организации обслуживания в устройстве с прерываниями используют пару блоков **PREEMPT (ЗАХВАТИТЬ)** – **RETURN (ВЕРНУТЬ)** так же, как для обычного устройства без прерываний использовались блоки **SEIZE** – **RELEASE.**

Блок **PREEMPT** имеет следующий формат (значение операндов приведено в табл. 26):

PREEMPT A,[B],[C],[D],[E]

Табл. 26. Значение операндов

Операнд	Значение	Результат по умолчанию
A	Имя устройства (числовое или символьное)	Ошибка
B	Возможность захвата по приоритету	Режим прерывания
C	Имя блока (числовое или символьное), в который переходит прерванный транзакт	
D	Номер параметра (числовое или символьное имя) у прерванного транзакта	
E	Возможность снятия с обслуживания	

Блок **PREEMPT** позволяет транзакту в зависимости от условий, заданных в операндах блока, занять устройство. Блок **PREEMPT** может также задержать транзакт на входе.

Операнд **A** определяет номер или имя устройства, на котором генерируется прерывание. Операнд может быть именем, положительным целым, СЧА или СЧА*СЧА.

Операнд **B** задает приоритетный режим (если **B** = **PR**) или режим прерывания (если этот операнд опущен). При работе в приоритетном режиме транзакт, уже занимающий устройство или генерирующий на нем прерывание, может быть прерван только транзактом, приоритет которого выше приоритета данного транзакта. Прерванные транзакты претендуют на дополнительное использование устройства, когда прервавший их транзакт войдет в соответствующий блок **RETURN.** Прерванные транзакты помещаются в список задержки в порядке приоритета.

Операнд **C** задает номер или имя блока, в который в этот же момент времени должен попытаться войти прерванный транзакт. Прерванный транзакт теряет управление устройством, но претендует на право его использования, если только не задан аргумент операнда **E**. В приоритетном режиме работы желательно задавать операнд **C**, если прерывающий транзакт имеет более высокий приоритет, чем прерываемый. Операнд может быть именем, положительным целым, СЧА или СЧА*СЧА.

Операнд **D** задает номер параметра, связанного с прерванным транзактом. Если прерываемый транзакт в момент прерывания направляется в список будущих событий, тогда остаток времени записывается в заданный параметр. Если такой параметр не существует, то он создается. В приоритетном режиме работы операнд **D** задают только в том случае, если прерывающий транзакт имеет более высокий приоритет, чем прерываемый транзакт. Операнд может быть именем, положительным целым, СЧА или СЧА*СЧА.

Операнд **E** задает либо не задает режим удаления (**RE**). В режиме удаления **RE** прерванный транзакт более не претендует на использование устройства и пытается войти в блок, заданный операндом **C** (если в операнде **E** стоит **RE**, то должен быть определен и операнд **C**). В приоритетном режиме работы режим **RE** используется только в том случае, если приоритет прерывающего транзакта больше приоритета прерываемого транзакта. При использовании **RE** прерванный транзакт не должен входить в блоки **RELEASE** или **RETURN,** связанные устройством, в котором обслуживался прерванный транзакт.

Если режим **RE** не задан (операнд **E** опущен), то прерванный транзакт по возвра-

щении в список текущих событий будет вновь пытаться занять устройство.

Прерываемый транзакт может находиться в списке будущих событий. Если надо сделать это, то используют операнд **D**.

Прерванный транзакт борется за устройство, даже если он перемещен операндом **C** (если **RE** не используется в операнде **E**). Если прерванный транзакт все еще борется за устройство, то попытка транзакта войти в блок **TERMINATE** приводит к ошибке. Такой транзакт перед входом в блок **TERMINATE** должен войти в блоки **RELEASE** или **RETURN.**

Транзакт может быть прерван на любом количестве устройств.

Устройство может быть захвачено любое количество раз, но не два раза подряд одним транзактом.

Транзакт не может войти в блок, если в приоритетном режиме устройство уже захвачено транзактом с приоритетом равным или большим, чем приоритет активного транзакта. Активный транзакт помещается в соответствии с приоритетом в список задержки устройства.

Транзакт не может войти в блок, если устройство находится в недоступном состоянии. Такие транзакты помещаются в список задержки устройства в соответствии с приоритетом, а внутри приоритета – по правилу FIFO.

Стандартные числовые атрибуты, связанные с описываемым блоком, те же, что и в табл. 19, с добавлением СЧА **FIj** – флаг прерывания устройства (1, если устройство находится в состоянии прерывания, 0 - в противном случае).

Следует обратить внимание, что при задании операндов **D** и (или) **E**, операнд **C** также должен быть задан.

Если приоритетный режим не задан (нет **PR** в операнде **B**), то операнды **C, D** и (или) **E** игнорируются. Однако возможен вариант, когда для прерванного транзакта выбирается альтернативный выход, причем приоритет транзакта не учитывается. Этот случай возникает тогда, когда задан операнд **C** (а иногда и операнды **D** и (или) **E**), но в операнде **B** не задан приоритетный режим. Такое использование операндов приводит к тому, что занимающий устройство транзакт прерывается и направляется по альтернативному пути. В данном случае многоуровневые прерывания не происходят.

Пары блоков **SEIZE** – **RELEASE** и **PREEMPT** – **RETURN** могут использовать одни и те же имена занимаемых устройств. В зависимости от логики работы модели пользователь должен сам определить, в каком случае разрешать прерывания, а в каком - нет.

Блок **RETURN** является парным к блоку **PREEMPT,** также как блок **RELEASE** к блоку **SEIZE,** и предназначен для освобождения ранее захваченного устройства. Он имеет следующий формат (значение операнда приведено в табл. 27):

RETURN A

Табл. 27. Значение операнда

Операнд	Значение	Результат по умолчанию
A	Имя устройства (числовое или символьное)	Ошибка

В операнде **A** задается номер устройства, с которого снимается прерывание. Прерывание может быть снято в блоке **RETURN** только тем транзактом, которым оно было сгенерировано.

Операнд **A** может быть именем, положительным целым, СЧА или СЧА*СЧА.

4.15. Сохраняемые величины

В GPSS пользователю предоставляется возможность определить «свои» глобальные переменные, начальные значения которых могут быть заданы перед моделированием и к которым можно обратиться из любого места модели в любой момент времени. Эти переменные называют сохраняемыми величинами (ячейками). Совокупность логически связанных между собой ячеек образует матрицу (аналог массива).

В отличие от параметров транзакта, приоритета и отметки времени, которые теряются в момент выхода транзакта из модели, ячейки доступны на протяжении всего процесса моделирования. Значения сохраняемых величин не подсчитываются интерпретатором автоматически (как СЧА устройств, очередей, МКУ и т.п.), а задаются и изменяются программистом.

Сохраняемые величины могут принимать положительные и отрицательные значения. Стандартный числовой атрибут **Xj (X$<имя ячейки>)** дает значение соответствующей сохраняемой величины. Например, **X2** - значение ячейки 2; **X$DAY** - значение ячейки **DAY**.

С матрицами связан стандартный числовой атрибут **MXj(m,n)** - значение, записанное в строке *m* и в столбце *n* матрицы *j* или **MX$<имя матрицы> (m,n),** если матрица имеет символьное имя.

СЧА сохраняемой величины может быть использован для косвенного задания данных, а также как аргумент функций и таблиц.

Перед использованием матрица определяется оператором описания **MATRIX.** Начальные значения ячеек и матриц можно задать с помощью оператора описания **INITIAL.**

Оператор INITIAL (ИНИЦИАЛИЗИРОВАТЬ). Если в процессе моделирования происходит обращение к сохраняемой величине, которая не была задана, то интерпретатор выдает ошибку в процессе выполнения программы. Поэтому перед началом моделирования все сохраняемые величины должны быть инициализированы с помощью оператора **INITIAL.**

Формат оператора представлен в табл. 28.

Табл. 28. Формат оператора

Поле	Информация в поле
Метка	Не используется
Операция	**INITIAL**
Операнд **A**	Имя сохраняемой величины
Операнд **B**	Начальное значение

Блок SAVEVALUE (СОХРАНИТЬ ВЕЛИЧИНУ). Значение сохраняемой величины изменяется при входе транзакта в блок **SAVEVALUE** (**СОХРАНИТЬ ВЕЛИЧИНУ**).

Его формат (значение операндов приведено в табл. 29):

SAVEVALUE A[+,-],B

Табл. 29. Значение операндов

Операнд	Значение	Результат по умолчанию
A	Номер или символьное имя сохраняемой величины	Ошибка
B	Величина, используемая для модификации (число или СЧА)	Ошибка

Подобно блоку **ASSIGN** блок **SAVEVALUE** может быть использован как в ре-

жиме замещения величины, так и в режиме увеличения или уменьшения. В режиме увеличения предыдущее значение сохраняемой величины увеличивается на значение, заданное операндом **B**, а в режиме уменьшения – уменьшается на это значение. Режимы увеличения и уменьшения определяются введением соответственно знака «плюс» или «минус» перед запятой, разделяющей операнды **A** и **B**.

Оператор описания матрицы MATRIX. Каждая матрица должна быть объявлена до ее использования, т.е. должна иметь оператор описания. Формат оператора представлен в табл. 30.

Табл. 30. Форма оператора

Поле	Информация в поле
Метка	Имя матрицы
Операция	**MATRIX**
Операнд **A**	Не используется (оставлен для совместимости с более старыми версиями GPSS)
Операнд **B**	Число строк матрицы (целое положительное)
Операнд **C**	Число столбцов матрицы (целое положительное)

Матрица в GPSS/PC может содержать не более чем 8191 элемент. Она может быть переопределена или инициализирована повторно другим оператором **MATRIX** с тем же именем. Переопределение, при котором размер матрицы изменяется, повлечет за собой выделение памяти под новую матрицу. Выделенная до этого оперативная память остается занятой.

Блок MSAVEVALUE. Блок **MSAVEVALUE** используется для записи значений в матрицы, а также для увеличения или уменьшения значений элементов матриц. Его формат (значение операндов приведено в табл. 31):

MSAVEVALUE A[+,-],B,C,D

Табл. 31. Значения операндов

Операнд	Значение	Результат по умолчанию
A	Имя матрицы	Ошибка
B	Номер строки матрицы	Ошибка
C	Номер столбца матрицы	Ошибка
D	Величина, используемая для модификации	Ошибка

Операнды **A, B** и **C** могут быть именем, положительным целым, СЧА или СЧА*СЧА. Операнд **D** может быть именем, СЧА или СЧА*СЧА.

Подобно блокам **ASSIGN** и **SAVEVALUE** этот блок может быть использован как в режиме замещения величины, так и в режиме увеличения или уменьшения.

Когда транзакт входит в блок **MSAVEVALUE,** то анализируется операнд **A** и ищется матрица с указанным именем. Если матрица не найдена, то возникает ошибка. Соответствующий элемент матрицы определяется содержимым операндов **B** и **C**. Если такого элемента не существует, то также возникает ошибка.

4.16. Проверка числовых выражений. Блок TEST

Сравнение СЧА может быть выполнено с помощью блока **TEST (ПРОВЕРИТЬ).** Его формат (значение операндов приведено в табл. 30):

TEST X A,B[,C]

Табл. 30. Значение операндов

Операнд	Значение	Результат по умолчанию
A	СЧА	Ошибка
B	СЧА	Ошибка
C	Имя блока, в который переходит транзакт при условии, что ответ на вопрос, подразумеваемый оператором отношения, отрицательный	При условии операнда **C** проверку выполняют в режиме отказа
X	Вспомогательный оператор, который представляет собой оператор отношения, использующийся при проверке	Ошибка

Значение оператора отношения	Вопрос оператора отношения
G	**A** больше **B**?
GE	**A** больше или равно **B**?
E	**A** равно **B**?
NE	**A** не равно **B**?
LE	**A** меньше или равно **B**?
L	**A** меньше **B**?

Операнды **A** и **B** – имена СЧА, которые сравниваются. Вспомогательный оператор **X** указывает способ сравнения этих двух СЧА друг с другом.

4.17. Определение и использование таблиц

Для накопления выборочных значений случайных величин и статистической обработки этих выборок используются GPSS-таблицы. Графическим аналогом GPSS-таблицы является гистограмма выборочных значений случайной величины, которую можно просмотреть в окне таблицы. Прежде чем использовать таблицу, ее нужно определить, а потом задать собираемые выборочные значения.

Оператор TABLE (ТАБЛИЦА). В модели может быть несколько таблиц. Каждую таблицу нужно сначала определить и только потом использовать в модели. Для определения таблицы необходимо указать:

1) имя таблицы (числовое или символьное);

2) имя случайной переменной, значение которой будет табулироваться;

3) число, являющееся первым граничным значением. (Значения выборки, меньшие или равные этому числу, попадают в самый левый (нижний) интервал (частотный класс) таблицы);

4) ширину интервала, общую для всех интервалов таблицы за исключением левого (нижнего) и правого (верхнего);

5) общее число интервалов таблицы, включая нижний и верхний. Формат оператора представлен в табл.31.

Табл. 31 Формат оператора

Поле	Информация в поле
Метка	Имя таблицы
Операция	**TABLE**
Операнд **A**	СЧА, значение которого учитывается в таблице
Операнд **B**	Первое граничное значение (целое число)
Операнд **C**	Ширина всех промежуточных интервалов (целое положительное число)
Операнд **D**	Общее число интервалов таблицы, включая левый и правый (целое положительное число)

На рис. 9 показана ось действительных значений и ее разделение на ряд интервалов таблицы.

Рис. 9. Ось действительных значений

Для сбора статистических данных об очередях используется оператор **QTABLE.** Его формат совпадает с форматом оператора **TABLE,** за исключением того, что операнд **A** задает имя очереди.

Блок TABULATE (ТАБУЛИРОВАТЬ). Выборочные значения попадают в таблицу в моменты вхождения транзактов в блок **TABULATE.** Его формат (значение операнда приведено в табл. 32):

TABULATE A

Табл. 32. Значения операндов

Операнд	Значение	Результат по умолчанию
A	Имя (символьное или числовое) таблицы, в которой табулируется соответствующий СЧА	Ошибка

Операнд **A** задает имя таблицы, в которую попадают выборочные значения. Одну таблицу можно использовать в нескольких блоках **TABULATE** модели. Отметим, что СЧА, по которому собирается статистика, в блоке **TABULATE** не указывается, так как он уже записан в операторе **TABLE.**

Стандартные числовые атрибуты таблицы:

TB<номер таблицы>, TB$<имя таблицы> - вычисленное среднее значение соответствующего СЧА;

TC<номер таблицы>, TC$<имя таблицы> - общее число входов в таблицу;

TB<номер таблицы>, TB$<имя таблицы> - вычисленное среднеквадратическое отклонение соответствующего СЧА.

4.18. Косвенная адресация

Ранее были рассмотрены прямые способы адресации, когда:

1) номер объекта задается константой

```
QUEUE     2  ; Стать в очередь 2
SEIZE     1 ;  Занять устройство 1
```

2) номер объекта задается СЧА

LEAVE P1 ; Освободить МКУ с номером, задаваемым параметром 1

Адресация может быть относительной и по отношению к самому блоку. Это осуществляется при помощи записи *+/–n. Например,

TRANSFER ,*+2

В этом случае сам блок **TRANSFER** является ориентировочным блоком. Транзакт пытается войти во второй, относительно блока **TRANSFER, блок.**

Подобные ссылки не зависят от свойств транзакта, обрабатываемого в данный момент времени. Использование прямой адресации может привести к введению большого числа дополнительных блоков только для того, чтобы записать номера объектов в СЧА, т.е. к увеличению объема модели. Существенным образом сократить объем модели и

использовать зависимость номеров объектов от свойств транзактов позволяет косвенная адресация.

Общий формат косвенной адресации:

C4A*C4Aj или **СЧА*СЧА<имя>**

Там, где допустимо использование СЧА, обычно могут использоваться и СЧА*СЧА.

При использовании косвенной адресации параметр **P** может опускаться.

При использовании косвенной адресации при обращении к устройствам, МКУ или функциям через параметры приходится применять числовые значения имен. Транслятор GPSS/PC на этапе компиляции сам присваивает именам числовые значения, однако, при этом пользователь не контролирует такое присвоение. Для того, чтобы не возникали конфликты между числовыми значениями имен, присвоенных транслятором, и именами, заданных пользователем, необходимо использовать простое правило: в модели использовать имена либо только числовые, либо только символьные.

Назначение именам числовых значений осуществляется с помощью оператора **EQU** (аналог **#define** в языке программирования Си).

Косвенная адресация является мощным инструментом языка GPSS, позволяющим существенно сократить размер модели и во многих случаях уложиться в ограничения для бесплатно распространяемых студенческих версий языка GPSS (в модели должно быть до 150 блоков).

4.19. Обработка транзактов, принадлежащих одному семейству

Кроме блока **GENERATE,** для создания транзактов может использоваться блок **SPLIT** (**РАЗДЕЛИТЬ**), который выполняет функцию копирования транзакта, входящего в него. Этот транзакт называется начальным или порождающим. Все копии формируются в момент входа начального транзакта в блок **SPLIT.** Каждая новая копия становится членом семейства (ансамбля) транзактов, порожденных одним начальным транзактом, который был создан блоком **GENERATE.**

Блок имеет такой формат (значение операндов приведено в табл. 33):

SPLIT A,[B],C]

Табл. 33. Значения операндов

Операнд	Значение	Результат по умолчанию
A	Число создаваемых копий транзакта	Ошибка
B	Метка блока, куда направляются копии	
C	Параметр, в котором запоминаются номера копий транзактов	

Операнд **A** может быть положительным целым, СЧА, СЧА*СЧА. Если вычисленное значение операнда **A** равно нулю, то блок **SPLIT** не выполняет никаких операций. После создания копий начальный транзакт пытается перейти к очередному блоку.

Операнд **B** задает блок, в который переходят копии начального транзакта. Операнд может быть именем (меткой), положительным целым, СЧА, СЧА*СЧА (в трех последних случаях операнд **B** задает номер блока). Значение операнда **B** вычисляется для каждой копии отдельно.

Операнд **C** задает параметр транзакта, который используется для присвоения копиям последовательных номеров. Операнд **C** может быть именем, положительным целым, СЧА, СЧА* СЧА.

Транзакты, принадлежащие одному семейству, объединяются интерпретатором в список. По связям внутри семейства транзактов невозможно установить, какой из транзактов семейства является назальным. Если копия транзакта входит в блок **SPLIT,** то повторная копия становится членом того же семейства, что и первичная копия. Таким образом, каждый транзакт является членом одного и только одного семейства. Семейство может состоять из произвольного числа транзактов. Когда транзакт уничтожается, интерпретатор автоматически исключает его из членов соответствующего семейства. Таким образом, семейство существует до тех пор, пока из модели не удалится последний из ее членов.

В модели одновременно может присутствовать произвольное вдело семейств, оно все время меняется, поскольку каждый транзакт, генерируемый блоком **GENERATE,** может создать свое семейство.

Для синхронизации движения транзактов, принадлежащих одному семейству, используются блоки **MATCH** (**СОГЛАСОВАТЬ**), **ASSEMBLE** (**СОБРАТЬ**), **GATHER** (**СОЕДИНИТЬ**).

Блок **MATCH** синхронизирует движение транзактов с другим блоком **MATCH.** Формат блока:

MATCH A

Операнд **A** указывает имя сопряженного блока. Сопряженным блоком является также блок **MATCH.**

При входе транзакта – сообщения в блок **MATCH** с меткой **LABEL1** он будет ждать (в списке синхронизации) момента, когда другой опросный транзакт, принадлежащий тому же семейству, не пойдет в сопряженный блок **MATCH** с меткой **LABEL2.** Только после этого сообщение займет канал **CHANNEL,** а опросное сообщение перейдет в блок **ADVANCE.**

Блок **ASSEMBLE** собирает начальный транзакт и все транзакты – копии из одного семейства, удаляет копии и выдает один начальный транзакт. После сборки из блока **ASSEMBLE** выходит только один транзакт, который переходит в следующий по номеру блок. Формат блока:

ASSEMBLE A

Операнд **A** задает счетчик сборки, указывающий сколько членов одного семейства должны быть объединены. Операнд **A** может быть именем, положительным целым, СЧА, СЧА*СЧА. Первоначальное значение операнда **A** не должно быть меньше или равно единице.

Блок **GATHER** скапливает заданное количество транзактов, принадлежащих одному семейству. Он задерживает их до тех пор, пока не соберется необходимое число, указанное операндом **A**. Затем накопленные транзакты одновременно попытаются войти в следующий по номеру блок.

Формат блока:

GATHER A

Операнд **A** задает число транзактов, принадлежащих к одному семейству, которое нужно накопить. Операнд **A** может быть именем, положительным целым, СЧА, СЧА*СЧА.

Для управления транзактами, принадлежащими к одному семейству, используется блок **GATE.**

4.20. Основные сокращения и обозначения симулятора GPSS

Основные сокращения и обозначения симулятора GPSS приведены в табл. 34.

Табл. 34. Основные сокращения и обозначения

Сокращение	Обозначение
TRANS	Номер транзакта
BDT	Время выхода сообщения из блока (это либо значение абсолютного условного времени, при котором транзакт покинет блок **ADVANCE**, либо значение абсолютного условного времени в момент, когда сообщение вышло из последнего блока **ADVANCE**, либо нуль, если сообщение еще не входило ни в один блок **ADVANCE**)
BLOCK	Номер блока, в котором сообщение находится в данный момент времени
PR	Уровень приоритета сообщения (0–127)
SF	Режим выбора (определяет режим выбора следующего блока: • пробел означает переход к следующему блоку; • "**A**" означает режим **ALL** блока **TRANSFER**; • "**B**" означает, что транзакт находится в блоке **TRANSFER**, работающем в режиме **BOTH**
NBA	Адрес следующего блока, в который должен войти транзакт
SET	Номер следующего транзакта того же семейства. (При создании транзакта в **SET** записывается номер самого транзакта. При входе в блок **SPLIT** в **SET** записывается номер следующего транзакта из образуемого семейства)
MARK	Отметка времени (записывается в момент создания транзакта или при входе транзакта в блок **MARK** с пустым полем **A**)
P1, …, P8	Текущие значения параметров 1–8
S1	Индикатор просмотра (при **S1** = 1 симулятор не будет пытаться продвинуть транзакт до измерения блокирующих условий; при снятии блокирующего условия **S1** устанавливается равным нулю для всех транзактов, задержанных данным условием)
T1	Индикатор трассировки (устанавливается в единицу блоком **TRACE** и в нуль блоком **UNTRACE**);
D1	Индикатор задержки для блока **TRANSFER SIM**
C1	Индикатор списка • **C1** = 0 – транзакт в списке пользователя, • **C1** = 1 – движение транзакта прервано, • **C1** = 2 – транзакт в списке текущих событий, • **C1** = 4 – транзакт в списке будущих событий;
MC	Индикатор синхронизации (если **MC** = 4, то транзакт находится в блоках **MATCH**, **ASSEMBLER** или **GATHER**)
PC	Счетчик прерываний
PF	Флаг прерываний.

5. Содержание пояснительной записки и требования к ее оформлению

Курсовая работа должна содержать:

1. выбор методов решения задачи;
2. классификацию системы;
3. построение концептуальной модели в виде *Q*-схемы;
4. блок-диаграмму системы;
5. аналитическую модель системы;
6. временную диаграмму, построенную для 5-10% требуемого объема моделирования;
7. текст (исходный код) программы полностью (часть программы, относящаяся непосредственно к алгоритму, должна быть подробно прокомментирована);
8. рассчитанные характеристики системы, указанные в задании;
9. выводы;
10. список использованных источников.

Список литературы

1. Веников В.А., Веников Г.В. Теория подобия и моделирования. М.: Высш. шк., 1984

2. Гнеденко Б.В., Коваленко И.Н. Введение в теорию массового обслуживания. М.: Наука, 1987

3. Клейнен Дж. Статистические методы в имитационном моделировании: [в 2-х вып.] М.: Статистика, 1978

4. Советов Б.Я., Яковлев С.А. Моделирование систем: Курсовое проектирование. М.: Высш. Шк., 2013

5. Советов Б.Я., Яковлев С.А. Моделирование систем.- М.: Высшая школа, 2012

6. Шрайбер Т. Дж. Моделирование на GPSS.