Ю.И. Рыжиков

ИМИТАЦИОННОЕ МОДЕЛИРОВАНИЕ

АВТОРСКАЯ ИМИТАЦИЯ СИСТЕМ И СЕТЕЙ С ОЧЕРЕДЯМИ

Учебное пособие

Санкт-Петербург
2018

УДК 004.94
ББК 32.973
Р

Рыжиков Ю.И.
Имитационное моделирование.
Авторская имитация систем и сетей с очередями:
Учебное пособие /Ю.И. Рыжиков. —
СПб.: Издательство «Лань», 2018. — 108 с.: ил.

ISBN

В книге описаны принципы и технология прямого имитационного моделирования на современном Фортране: базовые понятия имитации, техника генерации псевдослучайных чисел, программирование имитационных моделей, статистическая обработка результатов, методы понижения дисперсии.

Книга предназначена для специалистов в области исследования операций, связи, вычислительного дела, городского хозяйства и может быть использована для научно-исследовательских и проектных работ на стадии системного проектирования.

В вузах она может быть использована на всех уровнях (бакалавриат, специалитет, магистратура и аспирантура; переподготовка инженеров) при изучении курсов теории массового обслуживания, исследования операций, системного анализа, компьютерного моделирования, а также смежных дисциплин.

УДК 004.94

Набор, верстка и корректура выполнены автором

Оглавление

Предисловие

Многообразие приложений теории очередей определяет постоянно растущий интерес к ней, а сложность возникающих задач не позволяет получить исчерпывающие решения на базе аналитических методов [29, 32] даже при численной реализации последних.

В таких ситуациях приходится прибегать к *имитационному моделированию* (ИМ), представляющему собой незаменимый инструмент анализа эксплуатационных и многих других проблем. Имитационная модель помогает понять сложные системы, предсказать их поведение и развитие процессов в различных ситуациях; позволяет изменять параметры и структуру связей в системе, чтобы направить процессы в желаемое русло. Такое моделирование может осуществляться в *реальном времени*, что позволяет использовать его результаты в различных технологиях — от оперативного управления до тренинга персонала.

Интерес к ИМ в России за последнее десятилетие заметно возрос, о чем свидетельствуют сам факт проведения и материалы восьми всероссийских конференций [21, 24] и создание Национального общества имитационного моделирования, членом правления которого автор имеет честь состоять. Заметен и всплеск публикаций по ИМ [10, 13, 19, 22, 25, 36, 37, 38]. Подавляющая часть последних посвящена описанию работы в различных версиях GPSS и явно недостаточное внимание уделяет идеологии и принципам ИМ.

Фундаментальность высшего образования состоит в первоочередном изучении *основных идей* и *базовых принципов*: технологии должны осваиваться лишь по мере необходимости, по возможности и во вторую очередь. Это обстоятельство определило отбор материала для данной книги. Она написана на базе раздела лекций по моделированию систем, читанных автором в Военно-космической академии им. А.Ф. Можайского (первоначально в объеме главы из [5]), радикально переработанного в

1991 г. [27] и постоянно дополнявшегося с учетом современного состояния вопроса и потребностей прикладных исследований в течение последующих десяти лет [30]. Предполагается, что читатель знаком с основными понятиями теории массового обслуживания и фундаментальными соотношениями в ней типа формулы Литтла хотя бы в объеме третьей главы [29].

В данной книге расширены разделы о вариантах моделей систем и сетей обслуживания, многоагентном моделировании; добавлен материал по агрегативному моделированию и системе AnyLogic; обсуждается работа с распределением Парето. С другой стороны, ограниченность располагаемого учебного времени и требование фундаментальности заставили изъять подробные описания инструментальных (специализированных) систем моделирования, которые хотя и полезны в инженерной практике, но уяснению идей и принципов не служат.

Термин «прямая имитация» подразумевает разработку программ моделирования на алгоритмическом языке общего назначения — конкретно на современном Фортране [33]. Автор убежден, что работа в области прикладной математики без уверенного владения техникой программирования не только бессмысленна, но и опасна (и для общества, и для самого горе-специалиста). К сожалению, в последние десятилетия увлечение «информационными технологиями» и демагогия о «непрограммирующих пользователях» заметно ухудшили программистскую подготовку будущих инженеров и экономистов в их основной массе. В связи с этим в книгу включены типичные Фортран-процедуры имитационного моделирования — полностью или в виде фрагментов.

Автор стремился использовать традиционные и по возможности простые обозначения. В частности, большими буквами — например, (X, U) — обозначаются случайными величинами, а малыми (x, u) — их возможные значения. Малыми буквами с соответствующими нижними индексами обозначены моменты распределений. Если индекс опущен, то подразумевается первый момент. Малые буквы m и d зарезервированы для обозначения математического ожидания и дисперсии указанных индексом случайных величин, большие M и D — для обозначения соответствующих операторов. Знаком $\hat{}$ отмечены $оценки$ числовых характеристик — в отличие от их истинных значений. Поскольку в контексте данной книги все оценки случайны, применение $\hat{}$ однозначно характеризует объект как случайный и делает необязательным шрифтовое различие.

На протяжении всей книги используются следующие сокращения:

ТМО — теория массового обслуживания,
СМО — система массового обслуживания,
СеМО — сеть массового обслуживания.

Локальные обозначения и сокращения вводятся непосредственно в тексте. Англоязычное термины снабжаются переводом только при первом их упоминании.

Перечислим технические соглашения, принятые при написании текста. Предполагается, что буквенные клавиши нажимаются на нижнем регистре, т. е. без [Shift]. Если нужно нажать несколько клавиш одновременно, их имена заключаются в отдельные пары квадратных скобок и соединяются плюсом. Под обозначением вида [Ctrl] подразумеваются (в зависимости от контекста) как сама клавиша или кнопка, так и ее нажатие. Термины «кнопка», «клавиша» и «иконка» используются как синонимы. «Щелчок мышью» подразумевает нажатие левой ее кнопки.

Последовательный выбор команд из меню дается через слэш. Стандартные последовательности команд (наподобие выбора файла с типовой моделью) подробно поясняются при первом упоминании и — для лучшего запоминания — при нескольких последующих.

Набор и верстку книги автор выполнил в издательской системе MiKTEX. Соответственно пришлось мириться с заложенными в этой системе не вполне обычными решениями — в частности, с нумерацией сносок в пределах *главы*.

Глава 1

Базовые концепции имитационного моделирования

1.1. Моделирование систем

Модель есть материально или теоретически сконструированный объект, который заменяет (представляет) объект исследования в процессе познания, находится в отношении сходства с последним (аналогия, физическое сходство и т. п.) и более удобен для исследования. Изучение модели и выполненные над ней операции позволяют получить информацию о реальном объекте исследования и оптимизировать последний. Примерами реальных ситуаций могут служить работа крупного аэропорта [3] или железнодорожного узла; диспетчеризация тампонажных работ при бурении нефтяных и газовых скважин.

В сложной системе причинно-следственные отношения событий не являются простыми и ясными. Люди неспособны предвидеть результаты их взаимодействия. Работа с моделью помогает лучше понять системы, предсказать их поведение и развитие процессов в различных ситуациях; позволяет изменять параметры и структуру связей в системе, чтобы направить процессы в желаемое русло. Такое моделирование может осуществляться *в реальном времени*, что позволяет использовать его результаты в различных технологиях — от оперативного управления до тренинга персонала. Экспериментировать в этих целях с реальным объектом бывает неудобно, а зачастую вообще невозможно. Недопустимы,

например, эксперименты с экономикой страны, неосуществимы — с планетами Солнечной системы.

Модели дискретных событий определяют изменение состояний в *дискретные* моменты времени. В отличие от них, *непрерывное* моделирование отслеживает изменение состояний в непрерывном времени, на практике дискретизуемом при решении дифференциальных уравнений на ЭВМ. Типичные представители этого класса — модели системной динамики (макроэкономика, городской транспорт, динамика популяций и др.). *Гибридное* моделирование обычно включает аналитическую подмодель, вложенную в дискретно-событийную.

Наиболее естественная и важная область применения моделирования — анализ *систем*, позволяющий вести исследование в достаточно широком контексте. Система есть совокупность объектов, функционирующих и взаимодействующих друг с другом для достижения определенной *цели*. Понятие системы зависит от задач конкретного исследования. Ракеты, к примеру, суть компоненты *системы вооружения*, включающей не только саму ракету, но и стартовую установку, средства выявления и сопровождения цели, логистическую поддержку, оперативную обстановку, сведения о типовой тактике цели и т. д.

Состояние системы определяется как совокупность переменных, необходимая для ее описания и прогнозирования поведения в соответствии с задачами исследования. При анализе системы учитываются:

- общесистемные цели;
- окружение системы;
- компоненты системы (активные элементы, цели и показатели эффективности);
- ресурсы системы;
- управление системой;
- ограничения.

Все это в совокупности определяет структуру системы. *Цель* может быть сформулирована только с позиций вышележащего уровня.

При разработке новой системы для начала строится и оптимизируется стандартными методами *простейшая модель*, учитывающая важнейшие факторы. В любом случае моделируются все те и только те стороны процесса, которые влияют на выбранный показатель эффективности

или критичны к наложенным ограничениям. В процессе исследования модель постепенно усложняется за счет учета все новых и новых факторов, и ставятся задачи оптимизации по все большему числу параметров. Решение каждой из них обычно оказывается хорошим начальным приближением для следующей. Выполнение как общих, так и специфических для предметной области *законов сохранения* является важнейшим средством контроля корректности модели.

«Точная» модель как правило имеет бóльшую непосредственную ценность, но в узкой области применения. С другой стороны, исключение из обобщенной модели второстепенных факторов может поставить вопрос о применимости полученных результатов. Принцип *баланса точности* требует соизмеримости погрешностей, вызываемых различными причинами: неполным соответствием модели объекту, неточностью задания исходных параметров модели, случайным характером результатов моделирования.

При исследовании сложных систем может потребоваться разработка *набора моделей*, соответствующих различным иерархическим уровням рассмотрения и функциональным разрезам деятельности системы. Такое *стратифицированное* описание на каждом уровне использует свой набор концепций, понятий и терминов. Разработка иерархии моделей для предотвращения пропусков и бросовых затрат должна вестись «сверху вниз».

Математическому моделированию в любой его форме предшествует создание *содержательной* (концептуальной) модели, определяющей объект, цель и условия моделирования.

1.2. Общая характеристика имитационного моделирования

Аналитическими методами может быть исследован сравнительно узкий круг задач массового обслуживания. С другой стороны, уникальность и дороговизна многих реальных систем, а также ответственность решаемых ими задач мешают проведению натурных экспериментов — особенно в критических режимах (вспомним Чернобыль!).

При *имитационном* (=подражательном) моделировании реализующий модель алгоритм воспроизводит аналог процесса функционирования системы во времени и пространстве, причем имитируются

составляющие процесс элементарные явления с сохранением его логической (причинно-следственной) и временно́й структуры.

Чрезвычайно полезно и перспективно *комбинированное* использование аналитических и имитационных методов, позволяющее сочетать достоинства обоих подходов.

Растущий масштаб применения имитационного моделирования определяется как обострением потребностей в нем, так и облегчением разработки и применения моделей (рост быстродействия и удешевление компьютеров и создание программного инструментария, в том числе для визуализации процесса разработки и динамики моделей). Многие профессионалы исследования операций рассматривают ИМ как «последнее средство» решения сложной задачи.

Подчеркнем разницу между имитационным моделированием и методом статистических испытаний (Монте-Карло). Последний в общем случае не требует внешнего сходства с исходным процессом и не включает в себя явное представление времени.

В рамках *дискретно-событийной технологии* ИМ система исследуется путем прослеживания в модельном времени характерных событий, для каждого из видов которых создаются подпрограммы обработки событий. Эти подпрограммы подробно описывают изменения состояний, происходящие при наступлении события данного вида. Дискретно-событийные модели исключают из рассмотрения интервалы времени, не ознаменованные никакими событиями. Это уменьшает затраты машинного времени на прогон модели.

Процесс — это упорядоченная последовательность взаимосвязанных и разделенных промежутками времени событий, отражающая полный «жизненный путь» объекта в системе. Реализации процесса моделируются на ЭЦВМ с помощью серий случайных или псевдослучайных величин.

Управление работой модели строится в зависимости от цели исследования и организуется по текущему значению счетчика, связанного с этой целью.

Усреднение результатов моделирования по времени функционирования модели или числу реализаций процесса позволяет методами математической статистики получить *оценки* искомых характеристик.

Благодаря возможности достаточно полного отражения реальности (например, многокаскадного обслуживания, неоднородных потоков и каналов, блокировок из-за ограниченной емкости буферов, сложной

системы приоритетов и дисциплин обслуживания и т. п.) имитационное моделирование удобно для исследования практических задач: определения показателей эффективности, сравнения вариантов построения и алгоритмов функционирования системы, проверки устойчивости режимов системы при малых отклонениях входных переменных от расчетных значений и др.

Метод имитационного моделирования свободен от каких-либо ограничений на класс решаемых задач, нагляден, прост в освоении, обладает повышенной устойчивостью к сбоям ЭЦВМ; промежуточные результаты счета легко интерпретируются. Его целесообразно использовать:

- для накопления первичных данных об изучаемом явлении, если эти данные нельзя получить «в натуре»;
- для проверки правомерности допущений, сделанных разработчиком в целях перехода к аналитическим методам;
- для демонстрации конечных результатов исследования заказчику на достаточно полной модели реальной ситуации;
- при полной «безысходности», когда сложность ситуации намного превосходит возможности аналитических методов, известных разработчику[1].

Поскольку принципы ИМ общеизвестны и вполне элементарны, его применение в *квалификационных* работах нужно сводить к разумному минимуму.

1.3. Элементы имитационной модели

Имитационная модель образуется взаимодействием следующих элементов:

- состояний,
- событий,
- датчиков случайных чисел,
- таймера,

[1] Имеет смысл поинтересоваться возможностями неизвестных.

- цепей событий,
- цели моделирования,
- счетчиков,
- блока инициализации,
- критерия остановки,
- методов обработки результатов.

Состояние системы должно быть определено со степенью детальности, необходимой и достаточной для вероятностного продолжения процесса моделирования (процесс должен быть сведен к *марковскому*). В частности, состояние системы массового обслуживания задается текущим числом заявок в ней, фазами текущего обслуживания (прибытия) и моментами наступления ближайших событий каждого вида.

Под *событием* модели понимается скачкообразное изменение ее состояния. События могут быть первичными (прибытие заявки, завершение обслуживания) и вторичными (по отношению к завершению — накопление статистики, прием на обслуживание следующей заявки, продвижение очереди и т. п.), которые наступают как следствие первичных.

С помощью *датчиков случайных чисел* (ДСЧ) в модели формируются ее очередные состояния (моменты наступления следующих первичных событий каждого вида, объемы спроса на запасные части и т. д.). Случайные величины генерируются в соответствии с заданными распределениями.

Имитируемый процесс развивается в модельном (системном) времени. Счетчик модельного времени называется *таймером*. «Физическая» длительность имитации определяется уровнем детальности. При существенной разномасштабности процессов подробное моделирование нужно выполнять *отдельно* (снизу вверх) и полученные средние значения подставлять в модели более высокого уровня как константы либо переменные, зависящие от «медленных» параметров.

Модель выстраивает цепную последовательность событий в их взаимной зависимости. К неупорядоченной цепи проще добавляется новое событие, но после каждого добавления или выборки требуется ее просмотр для поиска наиболее раннего события. При вставке очередного события в список, упорядоченный по моментам наступления событий,

его целесообразно просматривать с конца или применять половинное деление.

Специализированные программные системы моделирования дискретных событий позволяют программировать модели в *потоковом* (процессном) стиле, описывая траектории прохождения заявок через устройства системы и указывая узловые точки для сбора статистики.

Под *инициализацией* понимается приведение модели до начала прогона в исходное состояние. Простейший аспект инициализации — обнуление всех накапливающих счетчиков. Для обеспечения воспроизводимости результатов (повторения прогонов) требуется установка в исходные состояния генераторов псевдослучайных чисел.

Цель моделирования при построении модели трактуется в узком смысле — как определение выбранных показателей качества функционирования системы. Выбор цели существенно влияет на структуру модели (через счетчики и операции, необходимые для накопления результатов моделирования). Упомянутые показатели обычно связываются с некоторыми *стационарными* (установившимися при устремлении системного времени к бесконечности) характеристиками. Данные, накопленные за время переходного к такому режиму процесса, будут вносить погрешность в окончательные результаты.

Критерий останова определяет момент прекращения прогона модели. В простейшем случае прогон прекращается по достижению заданного значения таймера, счетчика числа обслуженных заявок и т. п. Однако правильнее управлять прогоном по достижению заданной точности одного из определяемых показателей. Часто применяется двухэтапный прогон, когда на первом этапе грубо определяются величины, необходимые для формирования критерия остановки на втором этапе.

Обработка результатов моделирования состоит в сжатии получаемой информации, вычислении статистических (точечных и интервальных) оценок типа математических ожиданий и высших моментов для искомых показателей, оценке статистической значимости различия средних, построении гистограмм и статистических функций распределения.

1.4. Базовый алгоритм моделирования

На рис. 1.1 дается детальная структурная схема имитационной модели СМО типа $GI/G/n/R$, с помощью которой определяются стационарное распределение $p[0:R]$ числа заявок в системе и начальные моменты распределения времени ожидания начала обслуживания. В этой модели используются два класса событий: прибытие заявки и завершение обслуживания. Момент прибытия очередной заявки получаем добавлением случайного интервала к предыдущему, моменты освобождения каналов — добавлением к моменту начала обслуживания его случайной длительности. Упомянутые интервалы формируются посредством *датчиков псевдослучайных чисел*, настроенных на требуемые законы распределения. Приведем словарь этой модели:

`T`	—	таймер;
`T1`	—	момент предыдущего события;
`e[1:n]`	—	моменты завершения текущих обслуживаний;
`E`	—	наиболее ранний из них;
$\tau[n+1:R]$	—	моменты прибытия заявок в очередь;
`Z`	—	момент прибытия очередной заявки;
`p[0:R]`	—	счетчики стационарных вероятностей;
`r`	—	текущее число заявок в системе;
`ns`	—	общее число выбранных на обслуживание заявок (ограничено 5000);
`n1`	—	текущее число заявок в серии (ограничено 500);
`d`	—	количество отказов в приеме на обслуживание;
`s[1:4]`	—	счетчики для моментов распределения времени ожидания начала обслуживания;
`w[1:4]`	—	искомые моменты.

Управление прогоном производится по числу обслуженных заявок. Схема рассчитана на выполнение 10 прогонов с выдачей результатов в конце каждого из них.

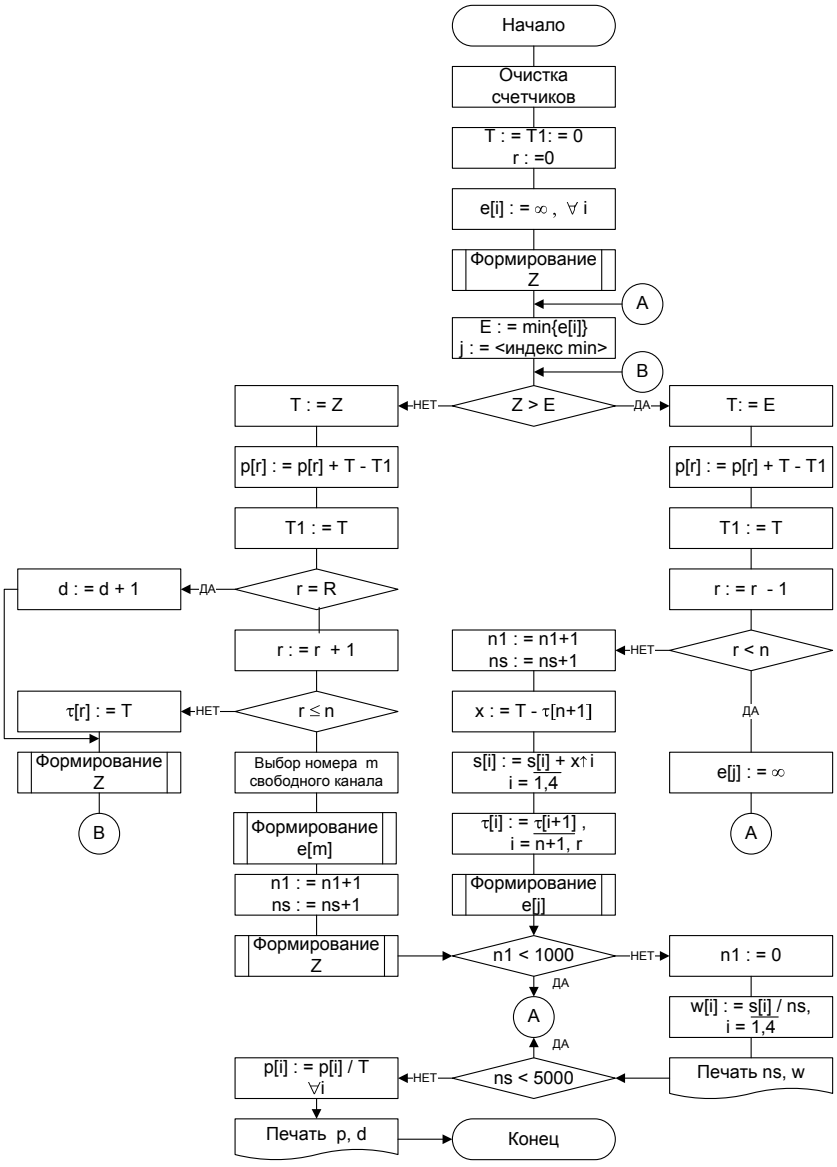

Рис. 1.1. Структурная схема имитационной модели

Работа модели начинается с установки в нуль таймера T (счетчика модельного времени), момента предыдущего события T1 и всех остальных накапливающих счетчиков. Моменты {e[i]} выхода из системы обслуженных заявок полагаем равными бесконечности (точнее — числу, заведомо превышающему предельное значение таймера в процессе прогона). Далее формируется случайный момент Z прибытия заявки и выбираются E (минимальный из моментов освобождения каналов {e[i]}) и индекс минимума j. Вариант обработки текущего события определяется соотношением между Z и E.

Если $Z < E$, то ближайшим событием в системе является прибытие заявки. Соответственно продвигается таймер T, к счетчику p[r] времени пребывания системы в состоянии с r заявками добавляется разность T-T1 и обновляется значение T1.

Если новая заявка застает в системе предельное число R ранее прибывших, то к счетчику отказов d добавляется единица. Затем формируется момент Z прибытия новой заявки. При наличии в системе свободного места r увеличивается на единицу и проверяется, есть ли свободные каналы. Если $r > n$, то их нет. Тогда для прибывшей заявки запоминается момент $\tau[r]$ прибытия; формируется очередная заявка. В обоих рассмотренных случаях мы выходим на метку B — сравнение E с обновленным Z.

При $r \leq n$ в системе имеется хотя бы один свободный канал. Тогда определяется его номер m[2]; в элементе массива e[m] формируется момент освобождения m-го канала; корректируются счетчики числа случаев ожидания n1 и ns (длительность ожидания в данном случае нулевая, и соответствующие счетчики времени ожидания не изменяются); опять же формируется момент прибытия новой заявки; при $n1 < 500$ происходит возврат к выбору минимального из {e[i]} (один из элементов этого массива обновляется).

В случае $Z > E$ очередным событием оказывается завершение начатого обслуживания. Здесь также продвигается таймер, пересчитывается счетчик p[r] времени пребывания в состоянии r, обновляется момент последнего изменения состояния T1. Затем число r заявок в системе *уменьшается* на единицу. Если $r < n$, то очереди в системе нет, в массив e для освободившегося j-го канала записывается «беско-

[2]Из свободных — канал с наименьшим номером, первый попавшийся, раньше всех освободившийся и т. п.

нечность» и выполняется переход на выбор минимальной компоненты массива e.

При r $\geq$ n в системе существует очередь, и в освободившийся канал будет выбрана на обслуживание новая заявка. Соответственно увеличиваются на единицу счетчики n1 и ns случаев выбора на обслуживание, вычисляется время x ожидания начала обслуживания головной заявкой очереди. Далее этот x и его последовательные степени добавляются к счетчикам {s[i]}, накапливающим данные для расчета моментов распределения времени ожидания. Затем продвигается очередь (точнее, записанные в ней моменты $\{\tau[i]\}$ прибытия в систему ожидающих заявок) и формируется время e[j] освобождения вновь занимаемого j-го канала.

При счетчике n1 $<$ 500 модель возвращается на выбор минимума из {e[i]}. При n1=500 прогон серии заканчивается, счетчик n1 обнуляется, рассчитываются и выводятся на печать оценки {w[i]} моментов распределения времени ожидания вместе с текущим значением ns. После достижения ns=5000 на печать выдаются стационарные вероятности {p[r]}, определяемые как доля общего времени T, в течение которого в системе находилось ровно r заявок, а также количество d отказов в приеме на обслуживание. На этом моделирование завершается.

1.5. Структуры данных

До сих пор мы рассматривали выбор заявок из очередей в порядке их прибытия, то есть по принципу FCFS (First Come — First Served). На обслуживание выбирается головная заявка очереди, после чего очередь *продвигается*. Новая заявка добавляется в конец очереди.

Для моделирования *случайного* выбора заявки из очереди текущая длина очереди Q умножается на случайное число $U \in [0,1)$. Результат (с добавлением единицы) определяет номер выбираемой заявки. Эта заявка выбирается в свободный канал, после чего очередь уплотняется (продвигается ее «хвост»).

При работе по правилу LCFS (Last Come — First Served) ожидающие обслуживания заявки образуют *магазин*. Здесь заявки добавляются в верхушку магазина и из нее же извлекаются. В обоих случаях необходим лишь *пересчет указателя магазина*.

Как обычную очередь, так и магазин можно организовать в виде

закольцованного массива длины M (закольцованность достигается относительной адресацией всех обращений к этому массиву по модулю M). В этом случае перемещение информационных элементов также заменяется пересчетом соответствующих указателей.

При составлении сложных моделей незаменимы *списки* — динамические наборы данных, каждый элемент которых состоит из информационных полей, хранящих целевые данные, и адресных, содержащих ссылки на смежные элементы. В отличие от очередей и таблиц, требующих резервирования постоянных объемов памяти «по максимуму», работа со списками позволяет добавлять и исключать элементы по необходимости — следовательно, использовать память по *реальной* потребности в ней. Применение списков не требует физического перемещения записей, что уменьшает трудоемкость моделирования при интенсивной динамике.

1.6. Элементы Фортрана 90

Читатель с опытом программирования без труда разберется в приводимом ниже тексте программы моделирования. Здесь мы отметим следующие особенности современного Фортрана:

1. Комментарий начинается восклицательным знаком.

2. Включаемая в ЦБС *запись* определяется как объект типа `event` с указанными типами значений полей. При ссылке на запись имя поля указывается через знак процента. Возможно объявление массива записей. При ссылке номер элемента указывается (в скобках) сразу за именем массива.

3. Динамические массивы, границы которых определяются в процессе счета, объявляются с атрибутом `allocatable`. Память под них после вычисления границ выделяется оператором `allocate` и по миновании надобности освобождается посредством `deallocate`.

4. Для конформных (с одинаковой протяженностью по всем измерениям) массивов разрешены групповые покомпонентные операции. Скаляр считается конформным любому массиву, что позволяет, в частности, обнулить любой массив одним оператором присваивания — без помощи цикла.

5. Вспомогательные внутренние процедуры включаются в конец охватывающей после ключевого слова `contains`. Объекты охватывающей процедуры для них являются глобальными.

Ниже приводится подпрограмма простейшей имитационной модели системы — без выдач промежуточных результатов моделирования, организуемых главной программой. Процедуры без параметров FTZ и FLIB формирования случайных интервалов между заявками и длительностей обслуживания должны оформляться как внешние функции и записываться в одном файле с BASEMOD и вызывающей ее главной программой.

```
 SUBROUTINE BASEMOD(N,JMAX,P,KMAX,THETA,NMAX)
!*****************************************
!*   Имитационная одель системы GI/G/N/R    *
!*       с FCFS- выбором из очереди         *
!*****************************************
      integer    n     !! Число каналов
      integer    jmax  !! Max число заявок в системе
      real*8     p     !! Стац. распред. числа заявок
      integer    kmax  !! Высш. порядок моментов ожидания
      real*8     theta !! Сами эти моменты
      integer    nmax  !! MAX число выборов на обсл.
      dimension  p(0:jmax),theta(kmax)

      integer          i,j,m,nref,ns,nz,r
      real*8           b,flib,ftz,s,told,tt,tz, &
                       tlib,ttlib,x,tau,
      dimension s      [allocatable] (:)
      dimension tau    [allocatable] (:)
      dimension tlib   [allocatable] (:)
      data      ns,nz,nref /3*0/

      allocate (s(kmax),tau(n+1:jmax),tlib(n))

      tlib=1d30               !! "Бесконечности" - по каналам
      ttlib=1d30              !!  Минимум из них
      p=0; s=0; r=0

      tt=0; told=0
      nz=0

      tz=tt+ftz()             !! Прибытие первой заявки
```

```
      do while(ns<nmax)       !! Продолжаем моделирование
         if (tz<ttlib) then
!*** Прибытие заявки ***
            nz=nz+1; tt=tz
            p(r)=p(r)+tt-told; told=tt
            if (r==jmax)  then
                nref=nref+1   !! Отказ
            else              !! Прием в систему
                r=r+1         !! Число заявок
                if (r<=n) then
!* Немедленный прием на обслуживание *
                   do i=1,n
                      if (tlib(i)==1d9) then
                          m=i    !! Номер свободн. канала
                          exit
                      end if
                   end do
                   tlib(m)=tt+flib(); ns=ns+1
                   call first(j) !! Выбор ttlib и канала j
                else
                   tau(r)=tt     !! Постановка в очередь
                end if            !! Приема заявки
            end if
            tz=tt+ftz()           !! Момент прибытия новой
         else
!*** Завершение обслуживания ***
            tt=ttlib
            p(r)=p(r)+tt-told; told=tt
            r=r-1
            if (r<n) then
!** Очереди не было **            !! Число заявок
                tlib(j)=1d9       !! Канал свободен
            else
!** Обработка очереди **
                ns=ns+1
!* Накопление моментов ожидания
                x=tt-tau(n+1)      !! Ожидание головной
```

```
               b=x
               do i=1,kmax; s(i)=s(i)+b; b=b*x; end do
!* Продвижение очереди
               do i=n+1,r; tau(i)=tau(i+1); end do
               tlib(j)=tt+flib() !! Занимаем канал j
            end if !!
           call first(j)
         end if                      !! Заявка/обслуживание
      end do                         !! Моделирования

      do i=1,kmax; theta(i)=s(i)/ns; end do
      do i=0,jmax; p(i)=p(i)/tt;  end do

      b=(1.0*nref)/nmax
      write (*,4) 'FRACTION OF REJECTIONS IS ',b
      deallocate (s,tau,tlib)
    4 format(30x,a,e14.5)

contains
      subroutine first(j)
!** Выбор ttlib = min из {tlib} и номера канала j
         integer j
         integer i
         ttlib=tlib(1); j=1
         do i=2,n
            if (tlib(i)<ttlib) then
                ttlib=tlib(i); j=i
            end if
         end do
      end subroutine first
end  subroutine !! basemod
```

1.7. Варианты моделей систем обслуживания

В этом разделе для каждого варианта обсуждаются только *впервые* возникающие в нем новые особенности.

Множественность типов событий. Для многовариантного разветвления (в нашем случае — множественности типов событий: прибытия заявки, завершения обслуживания, прихода тем или иным способом дезорганизующей обслуживание «отрицательной заявки», наступление меняющего режим работы системы «календарного события» — конец рабочей смены, наступление выходного дня, конец терпения заявки со случайным ограничением на время ожидания или пребывания в системе) можно применить оператор SELECT. Соответствующая конструкция открывается командой

```
SELECT CASE (<переменная>)
```

которой должно предшествовать присваивание этой переменной номера варианта, соответствующего типу ближайшего первичного события. Описания действий для вариантов начинаются операторами CASE (<номера-вариантов>). Весь оператор завершается командой END SELECT.

При большом числе типов событий целесообразно вести по ним раздельные цепи событий с указанием моментов наступления ближайших и *сводную* — содержащую эти ближайшие с указателем минимума из них. Иерархия цепей существенно сокращает объем операций по определению очередного события. Она особенно эффективны при моделировании *сетей обслуживания*.

Неоднородный поток заявок. Для многих сложных систем характерен неоднородный поток заявок. Например, при моделировании работы аэропорта [3] прибывающие самолеты различаются по типу, скорости, курсу, высоте полета, длине пробега и др. Здесь возможны два варианта. В первом из них необходимо иметь независимые указатели времени поступления заявок каждого типа и для сравнения с моментом окончания обслуживания отдельно хранить наиболее ранний из них. После приема в систему очередной заявки некоторого типа вырабатывается (при рекуррентных потоках) время прибытия новой заявки того же типа, выбираются новый минимум и индекс соответствующего источника. Во втором варианте моделируется поступление заявок *суммарного* простейшего потока интенсивности $\Lambda = \sum_i \lambda_i$ с дополнительным разыгрыванием типа заявки на основе вероятностей $\{\lambda_i/\Lambda\}$.

При неоднородном потоке необходимо либо помнить индекс типа заявки до ухода ее из системы, если заявка находится в общей очереди, либо отдельно строить очереди разнотипных заявок. Тип заявки учитывается:

- при постановке ее в очередь;
- при формировании случайной длительности обслуживания;
- при определении допустимого времени ожидания (пребывания в системе);
- при дифференцированной по классам заявок обработке результатов (подсчет числа отказов в обслуживании, моментов распределения времени пребывания в СМО и т. п.).

В подобных случаях на каждую заявку в момент ее прибытия в систему заводится $паспорт$, содержащий необходимую стартовую и оперативную (динамически изменяемую) информацию — например, оставшееся время обслуживания для прерванной заявки. Паспорт представляет заявку при всех ее перемещениях.

Многоресурсное обслуживание. Иногда для начала обслуживания заявки требуется одновременное наличие ресурсов нескольких видов, имеющихся в ограниченном объеме. В подобных случаях возможна ситуация типа известных из теории операционных систем «смертельных объятий» (deadlocks): несколько заявок из очереди частично обеспечили себя, захватив ресурсы разных видов. Они взаимно блокируются, что приводит к параличу процесса обслуживания. Для предотвращения таких блокировок можно исключить назначение неполных комплектов или ввести приоритеты выделяемых ресурсов. Такие задачи с большим трудом решаются численными методами и сравнительно легко — имитацией.

Кольцевая система очередей. Здесь очереди с независимыми источниками заявок обслуживаются «по кругу» — например, согласно одному из следующих вариантов:

- по одной заявке из каждой непустой очереди;
- до исчерпания заявок, скопившихся в текущей очереди к моменту начала ее обслуживания;
- до исчерпания текущей очереди.

Дополнительных рекомендаций требует только второй вариант: следует запоминать момент начала обслуживания текущей очереди и сравнивать его с моментами прибытия находящихся в ней заявок. Для выбора следующей очереди после i-й при $i = n$ принимается $i = 1$.

Статический относительный приоритет. Предпочтительна организация отдельной очереди по каждому классу приоритета. При завершении обслуживания очередная заявка выбирается из головы непустой очереди с наивысшим приоритетом. Обобщением этой стратегии являются *альтернирующие* приоритеты: текущая очередь обслуживается до ее полного исчерпания.

Динамический приоритет. В случае динамических (переменных во времени) приоритетов обслуживание заявок обычно ведется без прерываний, а диспетчерские приоритеты заявок растут по времени ожидания — как правило, линейно с коэффициентами $\{\beta_j\}$ и стартовыми вкладами $\{\gamma_j\}$, убывающими по мере понижения базового приоритета. Здесь следует хранить неупорядоченную общую очередь, а в паспорте каждой заявки фиксировать ее тип и момент прибытия. При освобождении канала для всех заявок очереди вычисляется диспетчерский приоритет и при необходимости обновляются значение и позиция текущего максимума. По окончании просмотра заявка с максимальным диспетчерским приоритетом выбирается на обслуживание, а следующая за ней часть очереди сдвигается вперед.

Многоуровневое квантованное обслуживание. Здесь появляется новый тип события — исчерпание кванта текущего обслуживания. Соответственно прерванная заявка перемещается в конец следующей, менее приоритетной очереди, где она будет дожидаться предоставления большего кванта.

Абсолютный приоритет, одноканальная система. Случайная длительность обслуживания для новой заявки генерируется в момент ее прибытия. Тип вновь прибывшей заявки сравнивается с типом обслуживаемой и определяется необходимость прерывания. Прерванная заявка помещается в голову очереди своего приоритета с потребным временем обслуживания, равным разности между расчетным моментом завершения его и текущим значением таймера (режим дообслуживания).

Многоканальные приоритетные системы. При моделировании *многоканальных* систем необходимо учитывать число и номера занятых каналов, а также возможную их дифференциацию по типам принимаемых заявок и характеристикам распределения времени обслуживания. Учет моментов освобождения каналов ведется аналогично обработке моментов поступления заявок неоднородного рекуррентного потока общего вида. При однородных каналах простейшей дисциплиной выбора занимаемого канала является «первый свободный», выявляемый по

«запредельному» моменту его освобождения.

При нескольких свободных каналах может потребоваться равновероятный выбор из них. Если таких каналов m, то для определения выбираемого канала следует вычислить $N = [m * U] + 1$, где U — случайное число, равномерно распределенное на полуинтервале $[0, 1)$, а квадратные скобки означают взятие целой части. Далее начинается просмотр моментов освобождения каналов, и для каждого свободного в счетчик добавляется единица. Заявка принимается в тот канал, для которого содержимое счетчика сравняется с N.

Особенно сложную логику имеет моделирование *многоканальных* систем с абсолютным приоритетом (эта задача весьма актуальна ввиду отсутствия аналитических методов ее решения в достаточно общей постановке). Хранение паспортов заявок, находящихся в системе, здесь целесообразно организовать в массиве `R` $\times$ 4, где `R` — предельное число заявок. Паспорт включает в себя тип заявки `z` (он же индекс приоритета), время прибытия в систему τ, требуемое время обслуживания θ. Для прерванной заявки в качестве θ записывается остаток времени обслуживания, вычисляемый как разность моментов планового завершения обслуживания и прерывания его. Четвертой компонентой паспорта для заявок, находящихся на обслуживании, является номер `C` занимаемого канала. Необходимость быстрого поиска прерываемых и выбираемых на обслуживание после освобождения канала заявок диктует упорядоченность списка паспортов по убыванию приоритетов, а при равных приоритетах — по возрастанию времени прибытия в систему. Первые n позиций списка заняты обслуживаемыми заявками. В этом случае приоритет вновь прибывшей заявки достаточно сопоставить с обслуживаемой в последнем канале. Такая технология требует переупорядочения списка при прерывании и при выборе заявки из очереди после завершении обслуживания.

Обозначим через π младший приоритет текущего обслуживания и сравним его с типом i новой заявки. При $i < \pi$ заявка, находящаяся в n-й позиции списка, прерывается; по номеру канала определяется остаток обслуживания; прерванная заявка помещается в очередь. При $i \geq \pi$ в очередь помещается новая заявка. Для быстрого поиска места вставки целесообразно иметь массив указателей начал «частных очередей» каждого приоритета. Место вставки для прерванной заявки нужно искать в начале частной очереди, а для новой — в конце. Если общая очередь уже имеет максимальную длину, то (в зависимости от соотношения приорите-

тов) получает отказ либо новая заявка, либо последняя заявка очереди. В паспорт прерывающей заявки заносится номер освобожденного канала, после чего для нее определяется позиция вставки. Затем выполняются вставки, сдвигаются нижележащие паспорта и корректируются указатели частных очередей. Заявка, выбираемая из очереди после завершения обслуживания, всегда помещается в n-ю строку списка паспортов.

Определение периода непрерывной занятости. В теории систем с приоритетным обслуживанием важную роль играют периоды непрерывной занятости (ПНЗ) системы заявками данного и более высоких приоритетов — на это время система недоступна для обслуживания заявок более низких приоритетов. При прибытии в n-канальную систему заявки, увеличивающей их текущее количество до n, фиксируется начало ПНЗ — значение таймера. При завершении обслуживания и уменьшении текущего числа заявок до $n-1$ отмечается окончание ПНЗ и к счетчикам добавляются последовательные степени его длительности (для вычисления моментов). К счетчику числа ПНЗ добавляется единица.

Учет временны́х ограничений. При моделировании системы с *временны́ми ограничениями* добавляется новый тип событий — «исчерпание терпения». Каждую заявку должен сопровождать допустимый момент начала (завершения) обслуживания. Для минимизации числа уходов по нетерпению имеет смысл упорядочить очередь по возрастанию упомянутых моментов. Соответственно для вновь прибывшей заявки придется определять место вставки и сдвигать часть очереди. Потеря «нетерпеливой» заявки должна сопровождаться уплотнением очереди и выбором следующей по «степени нетерпения». Такой выбор производится и в тех случаях, когда «нетерпеливая» заявка ушла из очереди в канал (при ограничении на время ожидания), при завершении обслуживания (ограничение на время пребывания), а также при прибытии новой заявки.

Цели моделирования и счетчики. Логика работы модели может меняться в зависимости от *целей исследования*. При расчете распределения стационарных вероятностей состояний в модели должны иметься счетчики по числу возможных состояний, и к их содержимому при каждом изменении состояний должно прибавляться время, проведенное в предыдущем состоянии.

Для расчета распределения числа заявок перед прибытием очередной заявки при прибытии новой заявки добавляется единица к счетчику,

соответствующему текущему (старому) числу заявок в системе. В конце моделирования содержимое всех этих счетчиков делится на полное число прибывших заявок. Аналогично могут быть определены вероятности состояний на моменты завершения обслуживания.

Для вычисления временны́х характеристик (занятости канала, пребывания заявки в системе) должны быть предусмотрены фиксация начала и конца соответствующих интервалов, а также накапливающие счетчики. Если требуется вычислить несколько статистических моментов случайной величины, счетчики необходимы по числу моментов.

При построении статистического распределения некоторой непрерывной случайной величины следует разбить возможный диапазон ее изменения на отрезки (обычно 15-20) и завести на каждый отрезок отдельный счетчик. Обработка содержимого этих счетчиков дает гистограмму распределения, а сопоставление ее (например, по критерию χ^2) с постулируемым теоретическим распределением позволит оценить приемлемость последнего. Статистические моменты можно непосредственно применить для апроксимации непрерывной плотности распределения.

1.8. Моделирование нестационарных процессов обслуживания

Значительная часть проблем теории очередей относится к *нестационарным* режимам. Здесь мы прежде всего отметим *периодические* режимы. Могие задачи управления городским хозяйством и социальной сферой связаны с суточными ритмами (потоки самолетов, пассажиров, пациентов, дорожно-транспортные происшествия, работа энергосетей). Годовой ритм имеют процессы развития ОРВИ- и желудочных заболеваний, лесных и торфяных пожаров, наводнений, лавин на горных склонах, обслуживания туристов, ледовой проводки караванов, обработки судов в северных морских и речных портах. *Непериодическими* нестационарными процессами являются двусторонние боевые действия (морской бой, ракетная дуэль), оказание медицинской помощи в ходе масштабной боевой или антитеррористической операции, борьба с эпидемией, аварийно-спасательная деятельность при землетрясении (в частности, с учетом «афтершоков») и т. д. Численно-аналитические методы расчета подобных ситуаций известны только для системы M/G/1, чрезвычайно сложны и трудоемки.

Обсудим возможную технологию имитационного решения поставленной проблемы. Прежде всего очевидно, что результаты моделирования должны определяться применительно к заранее заданным временным сечениям с номерами $k = \overline{1, K}$, последнее из которых совпадает с граничным значением таймера. Поскольку моделирование идет по схеме «от события к событию», окончание очередного отрезка должно связываться с первым перескоком таймера через его правую границу — лучше всего при наступлении события, связанного с накоплением статистики выбранного вида. В типичном случае нахождения распределения времени ожидания такими моментами могут быть $окончания\ ожидания$, которые связаны с приходом в недогруженную систему очередной заявки или с выбором заявки из очереди — при завершении обслуживания. После окончания отрезка содержимое «частных» счетчиков числа завершений и суммарного времени ожидания накапливается в «глобальных» счетчиках — по реализациям данного отрезка. Далее номер отрезка k увеличивается на единицу, а упомянутые «частные» счетчики обнуляются.

Конец прогона задается через предельное значение таймера — в момент наступления первого «знакового» события, превысившего этот предел. Возможный вариант — наступление «особого» события (снятие осады, конец вооруженного конфликта, взрыв или потопление корабля противника, стабилизация аварийной либо эпидемической ситуации и др.). Затем начинается новый прогон: устанавливается начальное состояние модели (в частности, обнуляется таймер) и номер отрезка $k = 1$. Количество прогонов (граница внешнего цикла) задается из стандартных статистических соображений. По окончании последнего прогона выполняется завершающая обработка — отдельно по каждому сечению.

Укажем некоторые особенности и варианты реализации алгоритма.

- Датчики временных интервалов и объемов заявок должны иметь аргументом текущее значение таймера, в зависимости от которого формируется длительность очередного интервала между заявками и/или обслуживания.

- В процессе продвижения заявки по сети ее должен сопровождать $паспорт$, в который могут заноситься данные, зависящие от таймера и влияющие на выбор маршрута в сети, длительность очередного обслуживания, предельный срок «терпения» и т. п.

- Возможны следующие схемы обработки окончания отрезка: обсуждавшаяся выше *интервальная* (характеристики, накопленные за интервал между смежными сечениями (например, средняя длительность ожидания помощи); *кумулянтная* (данные обрабатываются по ходу прогона нарастающим итогом); *мгновенная* (фиксируется «снимок» текущего состояния процесса: число больных, площадь пожара или затопленной территории, количество оставшихся целей).

Предложенная технология была проверена на модели системы M/M/2 – с простейшим входящим потоком и интенсивностью входящего потока, модулируемой синусоидой с относительной амплитудой M, равной 0.2 и 0.5. Среднее время обслуживания принималось равным $b = 0.25$. Максимальная интенсивность входящего потока выбиралась из условия $\rho = \lambda(1+M)\cdot b/2$. Значения максимального текущего коэффициента загрузки ρ задавались равными 0.9,1.0 и 1.1. Прогон общей длиной T=12 единиц делился на 12 интервалов. Круговая частота ω определялась из условия $\omega T = 2\pi$. Результирующий график, построенный с помощью системы `Gnuplot`, представлен на рис. 1.2.

Как и следовало ожидать, результаты моделирования показывают монотонный рост этих средних по максимальному коэффициенту загрузки. Поучителен кумулятивный эффект передачи очередей от сечения к сечению, вследствие которого ожидаемая синусоида реализовалась лишь в формате полуволны.

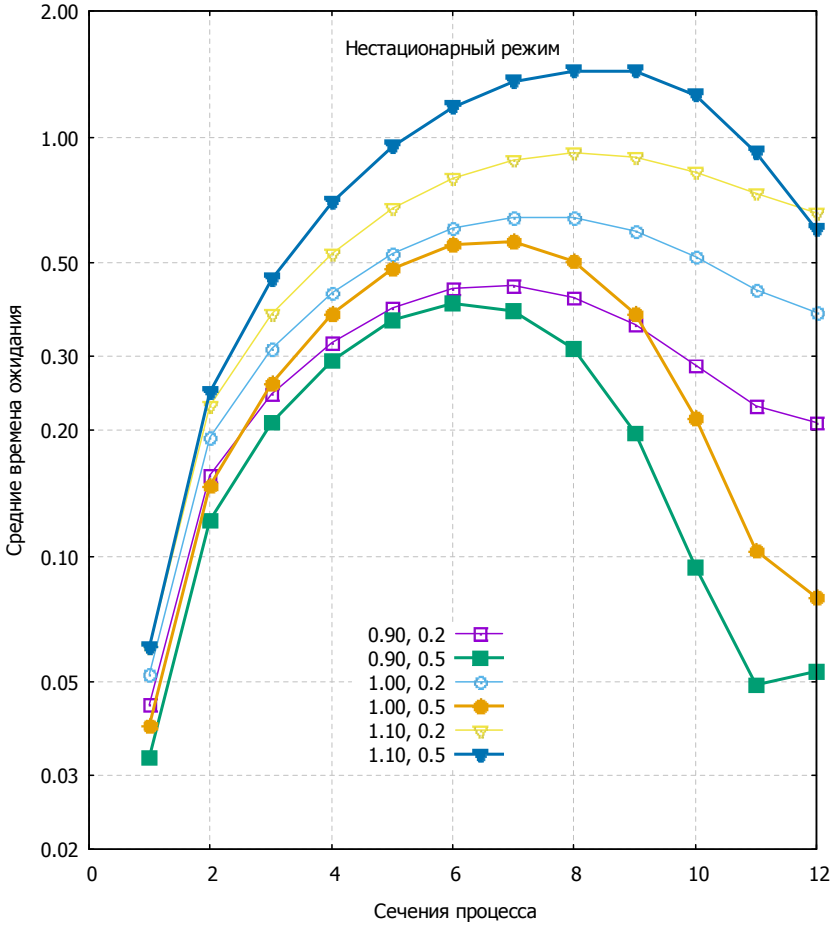

Рис. 1.2. Динамика изменения средних времен ожидания

1.9. Моделирование сетей обслуживания

Сеть обслуживания состоит из M рабочих узлов с n_i каналами обслуживания в каждом, $i = \overline{1, M}$, и двумя вспомогательными — нулевым (источником) и $(M+1)$-м — стоком. Вероятности перемещения заявки задаются матрицей передач $\{r_{i,j}\}, \quad i, j = \overline{0, M+1}$. Новая заявка рождается в источнике и адресуется в один из рабочих узлов с вероятностями $\{r_{0,j}\}$. *При всех перемещениях по сети заявку сопровождает*

ее паспорт, в котором содержатся отметка времени ее прибытия в сеть, индекс типа, требуемое (для прерванной заявки — оставшееся) время обслуживания и т. п. Заявка, попавшая в сток, считается покинувшей сеть.

Основными показателями работы сетей обслуживания являются моменты распределения времени пребывания заявки в сети и распределения числа заявок — в узлах либо в сети в целом. Время пребывания в сети каждой конкретной заявки определяется следующим образом:

1. Для входящей в сеть заявки формируется отметка времени ее прибытия.
2. На каждый j-й узел сети заводится отдельная очередь, в которой первые n_j (по числу каналов в узле) позиций хранят моменты прибытия в систему проходящих обслуживание заявок, а остальные — аналогичные моменты для заявок ожидающих.
3. Отметка времени при переходе заявки из узла в узел ее сопровождает.
4. При переходе заявки в сток определяется время ее пребывания в сети как разность между текущим значением таймера и отметкой времени прибытия. Вычисляются и накапливаются степени времени пребывания.

В целях подсчета маргинальных (частных) распределений числа заявок для каждого узла необходимо иметь указатель момента последнего изменения состояния узла и отдельный счетчик на каждое возможное состояние узла.

Сети обслуживания могут иметь весьма разнообразные особенности поведения, из которых мы относительно подробно рассмотрим две: меченые заявки и расщепление/слияние заявок.

В реальной сети обслуживания судьба заявки может зависеть от других находящихся в сети заявок. Считается, что эту зависимость можно нейтрализовать (точнее, статистически усреднить), если в сети в любой момент модельного времени находится не более одной «меченой» заявки. Соответствующая технология опирается на предложенный Е. Дийкстрой *метод семафоров*. В исходном состоянии сети семафор открыт (KEY=0). Входящая под открытый семафор заявка получает в свой паспорт специальную отметку и устанав-

ливает KEY=1. Для заявок, *покидающих сеть*, проверяется значение упомянутого признака. При его наличии к счетчику времен ожидания добавляется время, проведенное в системе данной заявкой, а к счетчику меченых заявок — единица. Затем семафор KEY вновь устанавливается в 0. Накопленные таким образом результаты касаются *только меченых заявок* — безотносительно к прочим. Естественная плата за этот результат — замедление процесса моделирования пропорционально среднему числу заявок в сети.

В сетях обслуживания часто встречаются процессы типа «Split and Join» — расщепления и слияния заявок. Примером может служить поступление в ремонт сложного изделия (автомобиля), блоки которого после разборки проходят по индивидуальным траекториям и затем поступают на сборку. Каждый блок при разборке получает двойной номер: исходной заявки и дополнительный индекс; для него определяется номер следующего узла сети. Затем продолжается работа стандартного алгоритма моделирования сети. В точке сборки накапливаются комплекты блоков. При достижении полноты одного из комплектов его компоненты аннулируются и запускается процесс сборки первоначальной заявки. Аналогично передаются сообщения по сетям связи с коммутацией пакетов.

1.10. Агрегативное моделирование

В 1960–1970-х гг. в Советском Союзе школой Н.П. Бусленко активно пропагандировался *агрегативный* подход [3], комбинирующий дискретно-событийное и « Δt »-моделирование. Предлагалось представлять систему как многоуровневую конструкцию из взаимодействующих элементов — кусочно-линейных агрегатов (КЛА). Последние функционируют как в дискретном «внешнем», так и в непрерывном «внутреннем» времени и включают как частные случаи конечные (в том числе вероятностные) автоматы, различные типы СМО и конечно-разностные схемы решения дифференциальных уравнений. «Внешними» считаются моменты поступления входных сигналов, а также выдачи выходных — при выходе состояния агрегата на границу допустимой области. Для описания агрегативной системы необходимо представить математические модели всех элементов и модель взаимодействия между ними.

В рамках этой концепции программа имитации есть последовательность макрооператоров, реализующих основные фрагменты работы КЛА: перемещение состояния внутри области, выход на границу, скачки основного состояния и дополнительных координат при выходе на границу и при поступлении входного сигнала, формирование выходного сигнала и т. п. Синхронизация системных событий в процессе моделирования осуществляется автоматически при помощи регистра, содержащего упорядоченные по возрастанию текущие значения опорных моментов всех элементов сложной системы. В последнем легко усмотреть аналог цепи будущих событий.

1.11. Моделирование вычислительных систем

Современные вычислительные системы являются не только незаменимым средством, но и важнейшим объектом имитационного моделирования, поскольку сложность протекающих в них процессов существенно ограничивает возможности применения к их расчету аналитических методов. Эта сложность порождается:

- иерархичностью построения вычислительных систем: узлы, блоки, устройства, машины, комплексы, локальные и глобальные сети — с возникновением на каждом уровне специфических проблем (здесь уместно вспомнить о многоуровневых протоколах взаимодействия открытых вычислительных сетей);
- сложностью алгоритмов их функционирования (достаточно сослаться на такие режимы, как конвейерная обработка, параллельные вычисления, квантованное обслуживание, режим реального времени);
- возможной потребностью заявки в одновременном использовании разнотипных аппаратных и программных ресурсов;
- сложностью и гибкостью операционных систем, развитие которых протекает значительно динамичнее совершенствования аппаратных средств и постоянно ставит все новые проблемы оценки временны́х характеристик, дополнительного расхода памяти, корректности

взаимодействия процессов, пропускной способности, надежности, защиты информации;

- необозримым разнообразием применений, со спецификой которых нельзя не считаться;
- включением в автоматизированные системы человека — в сочетании с трудностью формализации работы оператора АСУ;
- особой важностью задач, возлагаемых на ЭВМ в ряде случаев;
- трудностью прогнозирования социальных аспектов как штатных режимов вычислительных центров коллективного доступа, так и в особенности нештатных ситуаций (сошлемся на проблемы разграничения доступа, личных амбиций, хакерского зуда, «виртуальной реальности» и компьютерных вирусов).

1.12. Агентно-ориентированное моделирование

Главной чертой сложных социально-технических систем является наличие значительного числа субъектов, действующих относительно обособленно и имеющих возможность влиять на систему и другие субъекты путем принятия определенных решений. Динамика и функционирование агентных систем являются результатом индивидуальной активности членов групп.

Многоагентные системы (МАС) считаются одним из наиболее перспективных подходов к разработке сложных распределенных систем. Примерами проблемных областей для них являются логистика, мониторинг бизнес-процессов и состояния окружающей среды, ликвидация последствий природных и техногенных катастроф, электронная коммерция, сотовая связь, распределенное решение сложных задач, реинжиниринг предприятий и т. д. Агентно-ориентированный подход нашел применение в таких областях, как транспортные и телекоммуникационные сети, электронный бизнес, проектирование и др.

Определим базовые понятия агентного моделирования. *Агенты* — это автономные программные сущности, которые находятся в гетерогенной компьютерной среде и (совместно с другими агентами или в одиночку) стремятся к достижению определенных целей. Основа агента

является готовой компонентой, входящей в состав среды и используемой для генерации всех прикладных агентов. Она реализует набор проблемно независимых функций: запуск и остановка агента, отправка и получение сообщений, запуск и выполнение типовых сервисов, доступ к базе данных агента и внешним компонентам.

«Агентские убеждения» соответствуют представлениям агента о мире и могут быть неполными и/или некорректными. Агенты получают целевые установки (в общем случае не все они достижимы) от своего владельца (начальника), после чего могут функционировать автономно. Агент должен обладать достаточными способностями, чтобы в сфере своих задач замещать действия владельца, взаимодействовать с владельцем (пользователем) для получения задания и передачи результатов, ориентироваться в среде и принимать необходимые решения. Связь между агентами может обеспечиваться специальными «почтовыми агентами».

Предполагается, что агенты обладают следующими свойствами:

1) *наличие базовых знаний* о себе, о других агентах и об окружающей среде;
2) *целенаправленность* (собственная мотивация);
3) *убеждения*: переменная часть базовых знаний, изменяющаяся во времени;
4) *намерения*: действия, которые планируются агентом для выполнения своих обязательств и/или желаний;
5) *обязательства*: задачи, которые выполняет агент по просьбе или по поручению других агентов.
6) *активность*: способность к организации и реализации действий;
7) *реактивность*: агенты способны воспринимать внешнюю информацию и реагировать на нее;
8) *инициативность*: агенты могут выполнять определенные действия не только по запросам окружения, но и согласно своим планам и целям;
9) *автономность*: агенты функционируют без постоянного внешнего управления, осуществляют самоконтроль своих состояний и действий;

10) *мобильность*: способность мигрировать по сети в поисках необходимой информации, ресурсов и «сподвижников»;
11) *социальное поведение*: для достижения своих целей агенты взаимодействуют друг с другом посредством обмена сообщениями.

Иногда проводится дальнейшее «очеловечивание» агентов. Тогда к этому списку добавляют [1] такие свойства, как

- *правдивость*: неспособность к подмене истинной информации заведомо ложной;
- *благожелательность*: готовность к сотрудничеству с другими агентами (при отсутствии конфликта целей);
- *альтруизм*: приоритетность общего в сравнении с личным.

К базовым *видам взаимодействия* между агентами относятся сотрудничество (кооперация), конкуренция, компромисс, конформизм, изоляционизм (уклонение от взаимодействия). Аргументами за сотрудничество являются ограниченность собственных возможностей, привлечение необходимого опыта. Конкуренция порождается борьбой за ограниченные ресурсы. Взаимные обязательства позволяют предвидеть поведение других агентов и соответственно планировать собственные действия.

Формальное представление целей, обязательств, желаний, намерений и прочих характеристик агента составляют основу *ментальной модели* интеллектуального агента и определяют его поведение в автономном режиме. Агентские убеждения соответствуют представлениям агента о мире и могут быть неполными и некорректными. В общем случае не все цели агента достижимы.

Планы реализуют процедурные знания агентов. План имеет предусловия выполнимости, условия запуска и прекращения реализации. Тело плана может содержать цели (подцели) и действия (вызов релевантных процедур).

Особый интерес представляют МАС с элементами искусственного интеллекта — *когнитивные системы*. Такие МАС могут решать сложные задачи обработки информации и управления, связанные, в частности, с коллективным поведением роботов.

Событие, инициирующее выполнение подзадачи, как правило связано с выполнением коммуникационного акта согласно сценарию. Пусть, например, перед исполнителем роли `Manager` стоит задача назначить

работу и согласовать время ее выполнения. Manager посылает всем агентам, играющим роль Executor'а, сообщение с описанием требований в отношении данной работы. Каждый Executor, получивший такое сообщение, анализирует принятые им ранее обязательства и оценивает свои возможности выполнить предложенную работу. На основе этого Executor возвращает Manager'у либо отказ, либо предложение с описанием своих условий. Далее Manager анализирует полученные предложения, выбирает из них наиболее подходящее и посылает соответствующему Executor'у подтверждение своего согласия, а остальным участникам переговоров — отказ. Сценарий завершается тем, что выбранный Executor извещает Manager'а о своем согласии выполнить работу на оговоренных условиях.

Еще один пример –– агенты, разработанные для игровой среды *футбола роботов*. Агент-футболист имеет трехуровневую организацию поведения (технические навыки, индивидуальное поведение и поведение в команде). Все уровни поведения реализуются с помощью соответствующих баз знаний и средств логического вывода. В частности, предусматривается *обучение* игроков: агент действует так, чтобы максимизировать суммарное число поощрений, полученных за взаимодействие со средой в процессе обучения.

Разумеется, «робофутбол» является не конечной целью подобных исследований, но полигоном для отработки *общих* подходов к решению *реальных* проблем. Для примера сошлемся на кибернетическое противоборство в компьютерной сети, связанное с атакой типа «отказ по перегрузке». Такая ситуация представляется в виде противоборства команд программных агентов: злоумышленников и агентов защиты.

В [6] обсуждается применение мультиагентных систем к организации междугородних транспортных перевозок, управления такси, наземным сервисом аэропорта, внутрицеховыми операциями, грузопотоками международной космической станции, запуском ракет космического назначения, роем спутников.

1.13. Разработка имитационной модели

Рост масштабов и ответственности применений имитационных моделей породили повышенные требования к их корректности. Последняя не может быть добавлена к модели *после* ее разработки, но должна обеспечиваться на всем протяжении проекта. Здесь особую роль играют

начальные этапы. Допущенные на них ошибки исправляются с наибольшим трудом.

Типичный процесс моделирования проходит следующие фазы:

1. Организация.
2. Создание концептуальной модели.
3. Подготовка данных.
4. Программирование модели.
5. Верификация программы.
6. Планирование прогонов.
7. Машинный эксперимент.
8. Анализ результатов.
9. Интерпретация.
10. Реализация.
11. Документирование.

Перечисленные этапы в значительной степени перекрываются по времени (документирование должно вестись с первых дней работы над проектом) и охвачены многочисленными обратными связями.

1.13.1. Организация разработки

В *организацию* прежде всего входят:

- установление связей с заказчиком;
- определение задач;
- определение ресурсов (включая состав исполнителей);
- установление взаимодействия, отчетности и контроля;
- планирование работы.

По ходу всей работы над проектом проводятся:

- регулярное взаимодействие с заказчиком (уточнение постановки задачи, апробация допущений, поддержание интереса к проекту и чувства сопричастности, содействие внедрению);

- консультации с различными специалистами (проектировщики, пользователи, наладчики, системные администраторы, менеджеры и др.);
- наблюдение за системой (конкретные объекты наблюдения, единицы измерения, формат представления информации, требуемая точность, гарантии объективности);
- изучение опыта и инструментов решения подобных задач.

Следует подчеркнуть особую важность конструктивного взаимодействия с *заказчиком*: это ключ к достаточной реалистичности модели и успешному внедрению результатов ее исследования. В частности, весьма желательна работа над моделью в привычных для заказчика категориях и терминах.

Концептуальная модель включает в себя:

- обзорный раздел (общие цели проекта, конкретные проблемы, целевые показатели);
- предварительное описание подсистем и их взаимодействия;
- сделанные упрощения и их обоснование;
- располагаемые числовые данные и распределения;
- источники информации; оценка информации, в том числе по важности и непротиворечивости.

Определение системы предполагает уточнение ее границ с внешней средой; характеристики среды и внешних воздействий; изучение состава, назначения, внешних и внутренних связей; выявление ограничений и выбор показателей эффективности; постановку задачи на исследование. Описание представляется в виде схем, текстов, формул, таблиц, экспериментальных данных.

На этапе *формализации* строится математическая модель системы: устанавливаются ее структура и существенные зависимости между элементами. Здесь особенно важно выбрать модель минимально необходимой сложности. При большой сложности системы создается совокупность моделей по функциональному и/или иерархическому признаку. Показатели работы подсистем должны выводиться из целей системы в целом. Нужно иметь в виду, что при простом объединении для анализа системы в целом полных моделей подсистем нижних уровней возникает диспропорция между требуемой точностью и фактической сложностью

модели. Эта диспропорция может быть устранена загрублением моделей низшего уровня (после детального автономного исследования их). Возможными вариантами такого загрубления являются:

- укрупнение состояний и фаз процессов;
- аппроксимация выявленных зависимостей (например, кусочно-линейное представление функций распределения);
- усреднение характеристик процессов по их аргументам;
- сведение детальных описаний многокомпонентного процесса к главной составляющей с поправочными коэффициентами.

Далее обсуждаются принципиальные проблемы представления исходных данных. Здесь могут приниматься решения по таким вопросам, как

- стартовый уровень понятия «заявка» (детали, сборки, изделия, коробки изделий);
- типы распределений случайных величин;
- объединение выборок — по критериям однородности;
- характер потоков событий (стационарность, рекуррентность, ординарность — из общетеоретических соображений, качественного или количественного анализа ситуации);
- аргумент потока отказов (календарное время, наработка, число обслуженных заявок);
- методика расчета «штрафов» в случае отказов моделируемой системы, нехватки запасных частей и т. п.;
- основания для расчета стоимости хранения материальных запасов и организации их восполнения.

1.13.2. Подготовка исходных данных

Подготовка исходных данных состоит в сборе и обработке результатов наблюдений за моделируемой системой. Обработка в типичном случае заключается в построении функций распределения соответствующих случайных величин или вычислении числовых характеристик (моментов) распределений.

К подготовке исходных данных следует отнести и сбор информации о предполагаемых изменениях в нагрузке реальной системы (об ожидаемой нагрузке — для проектируемой системы).

Отчет о концептуальной модели должен быть понятен менеджерам. После рассылки его следует обсудить в кругу исполнителей, заказчиков и консультантов. При этом проверяются:

- полнота учета основных факторов и ограничений, влияющих на работу системы;
- соответствие исходных данных модели реальным (в частности, согласие постулируемых законов распределения с первичными данными).

Принятие положительного решения (с учетом поправок) рассматривается как *валидация* (признание ценности) концепции модели. Валидация концентрируется на установлении соответствия «реальности», тогда как верификация — на правильности разработки. Как валидацию *завершенного проекта в целом* можно рассматривать ретроспективную проверку ее прогностических возможностей.

1.13.3. Программирование и отладка модели

Трансляцией модели считается запись ее на одном из языков программирования — общецелевом или специализированном. Следует стремиться к блочному (модульному) построению программы, позволяющему независимо вносить изменения в отдельные модули и повысить производительность программирования путем повторного использования ранее созданных модулей (классов, объектов).

Проверка надежности программы модели, т. е. ее соответствия разработанной концепции, называется *верификацией* модели. Наиболее эффективным средством повышения надежности программ считается структурный подход к процессу программирования и жесткая (обязательно сертифицированная) технологическая дисциплина с соответствующей программной поддержкой.

Обсудим специфические особенности разработки и отладки программ *имитационного моделирования*. Прежде всего отметим сложность их логической структуры, выявление которой требует структурированной записи программ — следовательно, работы в свободном формате.

Промежуточные результаты имитационного моделирования, в отличие от результатов аналитического счета, всегда имеют четкий физический смысл. Это облегчает обнаружение ошибок в программе — в особенности при работе в интерактивном режиме.

При отладке необходимы режимы компилятора, выявляющие не описанные и не означенные переменные, а также выход индексов за границы массивов. В процессе отладки подлежат обязательной проверке:

- осмысленность результатов при нормальных условиях (поступательный ход модельного времени, отсутствие переполнения буферов и счетчиков, допустимый процент отказов в обслуживании) и в предельных случаях (пробные прогоны в условиях перегрузки системы обслуживания или работы с распределением Парето, *не имеющим конечной дисперсии*);
- выполнение для модели основных законов предметной области типа законов сохранения;
- правильность конечных результатов (например, при условиях, приводящих к известному аналитическому решению).

1.13.4. Прогоны и эксперименты

Планирование прогонов имеет целью получить при определенном объеме вычислительной работы для фиксированной точки пространства варьируемых параметров возможно лучшие статистические оценки показателей эффективности — несмещенные (в крайнем случае, асимптотически несмещенные) и с минимальной дисперсией. Может быть поставлена и обратная задача — получить оценки с заданной дисперсией при минимальном объеме работы. Отдельным прогоном (репликой) считается часть процесса имитации, в котором системное время монотонно возрастает.

Планирование экспериментов определяет совокупность исследуемых вариантов (в частности, наборов исходных данных) и/или стратегию их перебора. Здесь учитываются прежде всего:

- цель проекта (анализ или оптимизация);
- точность исходных данных (при малой точности необходимы дополнительные исследования чувствительности модели к вариациям параметров);

- ресурсы календарного и машинного времени.

При планировании работы с имитационной моделью полезно применение методов общей теории планирования экспериментов.

Регрессионный анализ служит для определения коэффициентов влияния изменяемых параметров и их комбинаций на результат (отклик) — по возможности на основании минимального числа прогонов (подробнее см. [30]). В тех же целях используется так называемый «пертурбационный анализ» — см., например, [46]. При *оптимизации* моделей обычно используются простые математические модели поверхности откликов.

Некоторые коммерческие пакеты моделирования содержат «оптимизационные» инструменты. В связи с этим отметим, что оптимизация сетей обслуживания фактически реализуется выбором типов обслуживающих устройств — из дискретного ряда поставляемых промышленностью, их количества (целые числа), дисциплин обслуживания, маршрутизации заявок и т. п. Получать эти компоненты решения стандартными методами *математического программирования* если и возможно, то нецелесообразно. Реальная альтернатива заключается в проведении экспериментов над моделью в *диалоговом* режиме с углубленным логическим анализом частных результатов (прежде всего — коэффициентов загрузки узлов) и первоочередной «расшивкой узких мест».

Как современные средства оптимизации компьютерных моделей используются также генетические алгоритмы, технологии искусственных нейронных сетей, эволюционные методы и др. Принцип действия *генетического алгоритма* заимствован у живой природы. Именно по такой схеме природа решает задачу формирования облика и характеристик живых существ, максимально приспособленных для их среды обитания. Такое приспособление осуществляется от поколения к поколению и реализовано природой при помощи механизмов скрещивания особей (кроссинговер), их мутаций и наследования потомками признаков их родителей. Чем более приспособленной является особь, тем больше потомков она воспроизведет. А особи-потомки унаследуют характеристики ее приспособленности. Некоторые потомки вследствие мутаций приобретут новые признаки, часть которых окажется полезной. Таким образом, суммарная приспособленность вида особей от поколения к поколению будет увеличиваться. Мера приспособленности особи служит показателем качества возможного решения.

Нейросеть (НС) есть набор связанных между собой нелинейных элементов — нейронов. Работа нейросети состоит в преобразовании входного вектора в выходной. Возможности сети возрастают с увеличением числа нейронов, количества связей между ними и числа внутренних слоев (каскадов). Настройка сети производится изменением интенсивности связей между нейронами. Этот процесс называется *обучением* НС и выполняется по определенным правилам, определяемым исходной информацией относительно объекта управления. При отсутствии эталона обучение проводится «без учителя» — с помощью генетических алгоритмов, в ходе реализации которых меняются упомянутые интенсивности.

1.13.5. Анализ результатов, интерпретация, реализация, аккредитация

Этап *анализа результатов* определяется спецификой исходного объекта и модели. Его важнейшим итогом является *верификация* — проверка соответствия логики работы модели требованиям задания.

Интерпретация результатов состоит в переносе их с модели на исследуемую (проектируемую) систему.

Реализация заключается в практическом осуществлении полученных рекомендаций и оценке их эффективности в процессе опытной эксплуатации.

Валидация модели есть признание ее практической ценности заказчиком.

На всех этих этапах тесное взаимодействие с заказчиком играет особую роль.

Тщательное и полное *документирование* процесса разработки модели и хода экспериментов дисциплинирует участников проекта; повышает вероятность его успешной реализации; обеспечивает накопление научно-технического опыта и обучение специалистов по моделированию; позволяет модифицировать модель в целях ее дальнейшего использования.

Аккредитация модели заключается в официальном признании компетентным органом ее пригодности или полезности для некоторого класса ситуаций и соответственно — в рекомендации модели к более широкому использованию.

Контрольные вопросы

1. Самостоятельно нарисуйте базовую схему имитационной модели. Убедитесь в ее правильности, «поползав» по ветвям алгоритма.
2. Модифицируйте базовую схему имитационной модели применительно к расчету моментов времени *пребывания*.
3. Как организовать равновероятный выбор свободного канала?
4. Как и когда формировать длительности обслуживания заявки для дисциплин с прерыванием PR, RS, RW?
5. Нарисуйте схему имитационной модели двухканальной системы с относительным приоритетом.
6. Нарисуйте схему имитационной модели одноканальной системы с заявками, нетерпеливыми в очереди. Определите среднюю частоту уходов в единицу времени.
7. Нарисуйте схему имитационной модели одноканальной системы с заявками, нетерпеливыми в системе. Определите среднюю частоту уходов в единицу времени.
8. Нарисуйте схему имитационной модели тандема из двух одноканальных систем с ограниченным буфером перед второй системой.
9. Нарисуйте схему имитационной модели циклической системы с квантованным обслуживанием.
10. Нарисуйте схему имитационной модели многоуровневой системы с квантованным обслуживанием.
11. Укажите особенности моделирования сетей обслуживания.
12. В чем состоит специфика моделирования вычислительных систем?
13. В чем сущность мультиагентного моделирования? Приведите собственные примеры мультиагентных ситуаций.
14. Опишите идею генетического алгоритма оптимизации.
15. Что такое валидация, верификация и аккредитация модели?

Глава 2

Генерация случайных чисел

2.1. Принципы аппроксимации распределений

Выравнивание статистических распределений, т. е. подбор теоретических зависимостей, описывающих фактически наблюдавшиеся данные, обычно проводится при условии сохранения значений интегральных числовых характеристик распределений, аккумулирующих его основные свойства. В качестве таких характеристик чаще всего выступают начальные моменты

$$f_i \stackrel{\text{def}}{=} \int_0^\infty t^i f(t)\,dt, \qquad i = 1, 2, \ldots$$

В некоторых случаях применяется подбор распределений по методу *квантилей* — исходя из сохранения значений ФР в заданных точках.

Ниже рассматриваются способы генерации *псевдослучайных* чисел, подчиненных требуемым законам распределения.

2.2. Схема получения случайных чисел

Моделирование СМО на ЭЦВМ требует большого количества случайных чисел (интервалы между заявками, длительности обслуживания, продолжительность ремонта отказавшего устройства, допустимые времена ожидания, запрашиваемые очередной задачей объемы оперативной памяти).

Применение «реальных» данных обеспечивает наилучшее приближение к фактически наблюдавшемуся процессу, однако при этом:

- не гарантируется *типичность* данных;
- длительность моделируемого процесса ограничивается длительностью реального;
- модель лишается прогностической силы, поскольку для проектируемых систем или измененных вариантов построения и использования реальных систем такие данные отсутствуют.

В практике моделирования почти исключительно применяются ДСЧ, *формирующие* случайные числа с нужным законом распределения.

Получение случайных чисел с требуемым законом распределения обычно выполняется в два этапа:

1. Формирование физическим или программным методом случайного числа U_i, $i = 1, 2 \ldots$, равномерно распределенного на полуинтервале $[0, 1)$.
2. Программный переход от U_i к случайному числу X_i, имеющему требуемое распределение $F_X(x)$.

Генераторы оценивают по качеству формируемой последовательности, быстродействию, трудоемкости инициализации, машинной независимости. Доверие к выработанной серии случайных чисел возрастает, если они прошли *специализированный* тест — упрощенный вариант модели исходной системы, для которой известно аналитическое решение. Таким тестом при решении задач ТМО может служить, например, хорошо изученная схема аналитического расчета системы типа M/M/1.

Обширные таблицы способов генерации различных распределений приведены в [4, 13]. Один из самых полных обзоров содержится в [44] (1986 г.). В [47] цитируется данный Д. Кнутом (1997 г.) свод техники генерации случайных чисел и тестирования генераторов.

В программах некоторых современных ДСЧ предусматривается генерация за одно обращение *массива* случайных чисел. Эффективное быстродействие датчиков здесь увеличивается из-за уменьшения числа обращений к подпрограмме.

2.3. Генерация равномерно распределенных чисел

2.3.1. Физические и программные датчики

Равномерно распределенное на $[0,1)$ случайное число представляется в ЭЦВМ в двоичной форме в виде n-разрядной последовательности нулей и единиц с десятичной точкой, фиксированной перед старшим разрядом. При этом в каждом разряде нуль или единица должны наблюдаться с вероятностью 0.5.

Физические датчики равномерно распределенных на [0,1) чисел состоят из n идентичных по своим параметрам триггеров со счетным входом, каждый из которых регистрирует независимый поток импульсов от счетчика радиоактивных частиц или шумовые выбросы электронной лампы. Такой поток можно считать простейшим. Неизбежная нестабильность физических датчиков требует регулярного аппаратурного и математического контроля *каждого экземпляра* датчика, существенно усложняя его математическую эксплуатацию.

Программные ДСЧ фактически генерируют *псевдослучайные* числа. Все ДСЧ этого класса обеспечивают близкое к равномерному перемешивание разрядов исходных чисел некоторым *закономерным* способом. Принято считать последовательность псевдослучайных чисел случайной, если «каждый ее член непредсказуем для непосвященного и она удовлетворяет ряду традиционных статистических тестов, в некоторой степени зависящих от цели выработки последовательности».

Программные ДСЧ имеют следующие преимущества:

- отсутствие дополнительного оборудования;
- возможность повторения прогона с той же последовательностью случайных чисел в целях контроля вычислений, уменьшения дисперсии результатов или сравнительного анализа вариантов;
- необходимость лишь однократной проверки ДСЧ после его разработки.

Работа ДСЧ оценивается по согласию статистического распределения $\{U_i\}$ с теоретическим — прежде всего по равномерности заполнения r-мерного единичного гиперкуба (при $r = 1$ — отрезка $[0,1)$,

при $r = 2$ — единичного квадрата). Наглядным средством визуального контроля для двумерного случая является равномерность засветки на экране единичного квадрата (координатами каждой точки является пара смежных чисел). Согласие фактических распределений с ожидаемыми устанавливается, например, по критерию χ^2. Дополнительно проверяется отсутствие корреляции между последовательными числами и между отдельными разрядами чисел. Используются также тест серий (длин подпоследовательностей элементов с определенными свойствами) и тест «максимумов и минимумов».

2.3.2. Идея построения программных датчиков

Ключевым элементом технологии ИМ является формирование псевдослучайных реализаций случайных величин на основе программных датчиков псевдослучайных чисел $\{U_i\}$, равномерно распределенных на интервале [0,1). Такие ДСЧ обычно реализуются некоторым закономерным пересчетом целых чисел, нормируемых к интервалу $[0, 1)$ делением на максимально допустимое целое число (модуль датчика) либо умножением на обратную ему величину.

Поскольку в n-разрядной сетке количество различных двоичных чисел равно 2^n (для машин с 32-разрядным словом $\approx 4 \cdot 10^9$), после прохождения некоторого стартового участка отрезки последовательности различных $\{U_i\}$ рано или поздно начнут *повторяться*. Тестирование программного ДСЧ должно быть произведено на сериях различной длины вплоть до длины периода (может получиться так, что тесты для периода в целом дают хороший результат, а для отдельных его частей — неудовлетворительный). Подобную проверку должна пройти каждая новая программа, предлагаемая к использованию на ЭЦВМ данного типа. Использовать в одной задаче количество чисел, превышающее длину отрезка апериодичности L, не рекомендуется. Поэтому программист должен знать длину периода для применяемой им программы ДСЧ.

Потенциальным недостатком программных ДСЧ является опасность *вырождения* — получения на некотором шаге чисто нулевого кода, последующие преобразования которого дадут опять же нули.

2.3.3. Классические конгруентные генераторы

Чаще всего применяют линейные мультипликативные генераторы вида

$$X_k = (aX_{k-1}) \pmod{M}, \qquad k = 1, 2, \ldots$$

где все операнды — целые и для n-разрядных чисел $M = 2^n$. Конечный продукт $U_k = X_k/M$. Рекомендации по выбору параметров таких генераторов приводятся, например, в книге Д. Кнута [15].

Если X_{k+1} зависит только от X_k, то длина периода не превосходит количества M различных чисел. В случае $X_{k+1} = f(X_{k-1}, X_k)$ максимальный период равен M^2. Поэтому лучшие результаты дают генераторы высших порядков вида

$$X_{i+n} = (a_1 X_{i+n-1} + a_2 X_{i+n-2} + \ldots a_n X_i) \pmod{M}.$$

Их частным видом являются *генераторы Фибоначчи*, например

$$X_k = X_{k-17} - X_{k-5}.$$

Работа с датчиками подобного типа предполагает их *инициализацию*: задание базовых целых значений и заполнение закольцованного стартового массива.

Наконец, возможно введение дополнительного перемешивания, при котором сначала заполняется таблица из M чисел. Далее генерируется целое число $i \in [0, M-1]$, выдается i-й элемент таблицы, а на его место записывается новое случайное число. Если выбрать, например, M =128, то можно на каждом шаге формировать только одно случайное число, а в качестве входа в таблицу использовать семь младших разрядов этого числа.

Популярна точка зрения, что при достаточном числе испытаний можно получить все требуемые результаты с *любой* точностью. Однако реальные датчики равномерных псевдослучайных чисел отнюдь не идеальны, и игнорировать это обстоятельство нельзя. В подтверждение покажем (таблица 2.1) динамику погрешностей трех моментов распределения $\{U_i\}$, точные значения которых равны соответственно $\{1/2, 1/3, 1/4\}$.

Таблица 2.1. Тестирование датчика равномерных чисел

Тыс. испыт.	Погрешности моментов			Погрешность
	f_1	f_2	f_3	π
2	-6.33e-3	-9.54e-3	-1.09e-2	4.84e-2
5	-3.03e-3	-4.06e-3	-4.71e-3	1.36e-2
10	-5.19e-3	-4.94e-3	-4.70e-3	-3.59e-3
20	-2.76e-3	-2.77e-3	-2.76e-3	-4.39e-3
50	-1.29e-4	3.81e-4	4.79e-4	1.93e-3
100	-1.11e-5	2.07e-4	2.16e-4	1.67e-4
200	2.58e-5	-7.15e-5	-1.95e-4	-2.93e-4
500	2.83e-5	-3.35e-5	-1.15e-4	2.47e-3
1000	2.26e-4	1.89e-4	1.14e-4	-7.45e-3
2000	-1.26e-4	-1.23e-4	-1.13e-4	-3.27e-4
5000	9.14e-6	2.91e-5	3.42e-5	5.44e-4
10000	9.17e-6	5.55e-6	-1.24e-6	-6.75e-5
20000	6.70e-5	7.31e-5	6.49e-5	-3.99e-4
50000	7.76e-5	8.00e-5	7.07e-5	-2.87e-4

В последней колонке приведены погрешности оценки π, определяемой через долю точек, попавших во вписанный в единичный квадрат круг (кстати, это поучительный пример статистических испытаний *без имитации*). Как мы видим, надежды на неограниченное увеличение точности увеличением количества испытаний более нескольких миллионов явно не оправданы.

Отметим, что пик исследований по ДСЧ остался в прошлом — когда работали на гораздо более слабых ЭВМ и число испытаний вынужденно ограничивалось десятками тысяч. «Дальние» зоны работы датчиков исследованы плохо. Поэтому рекомендуется отлаживать модель на задаче с известным решением (здесь могут помочь численные методы теории очередей), а затем подобрать датчики и число испытаний, при которых имитация обнаруживает наилучшее согласие с эталоном.

2.3.4. Раздельные датчики

Принципиальным условием корректного моделирования является взаимная независимость случайных величин каждого рода (для модели СМО — интервалов между заявками, длительностей обслуживания, типов заявок, номеров узлов-преемников в сетях обслуживания и т. п.).

Она достигается использованием для их генерации *раздельных* ДСЧ. Для удобства реализации датчики должны быть однотипны. Как правило это генераторы мультипликативного вида с общим множителем, но различными начальными значениями. Требуется гарантировать непересекаемость генерируемых серий.

Упомянутую проблему можно решить, если разделить период генератора (для хороших ДСЧ это 2^{m-2}, где m — разрядность датчика) на требуемое число серий и взять в качестве начальных значений числа с соответствующими номерами. Для получения этих чисел можно применить алгоритм ускоренного получения случайного числа, основанный на двоичном разложении его номера k:

```
subroutine fastrand(k,p)
   integer k,p
   integer j,j1,j2,x0,z
   data x0 /57539/
   j=k; p=x0
   z=1220703125
   do while(j>0)
      j1=j/2; j2=j-2*j1
      if (j2==1) then
         p=p*z
         if (p<0) p=(p+2147483647)+1
      end if
      z=z*z
      if    (z<0) z=(z+2147483647)+1
      j=j1
   end do
end
```

Здесь как очередной «полуфабрикат» p, так и степень множителя z копятся по модулю датчика 2^{31}. Возведение в степень выполняется на каждом шаге, а домножение p — в среднем на половине шагов, что дает для трудоемкости получения k-го числа оценку $\frac{3}{2}\log_2 k$ умножений. Заметим, что двоичный логарифм *миллиарда* не превосходит 30. Алгоритм применим к любым начальным значениям и множителям и (после очевидной модификации) — к любому модулю датчика.

При практическом использовании датчиков целые числа для получения $\{U_i\} \in [0,1)$ делятся на модуль датчика либо умножаются на обратную ему величину.

2.4. Метод обратной функции

2.4.1. Идея метода

Универсальным способом перехода к требуемому распределению $F(x)$ случайной величины является метод обратной функции. На рис. 2.1 показана его графическая реализация. Здесь U — случайное число, равномерно распределенное на интервале [0,1). Таким образом, из уравнения $F(X) = U$ определяется

$$X = F^{-1}(U). \tag{2.4.1}$$

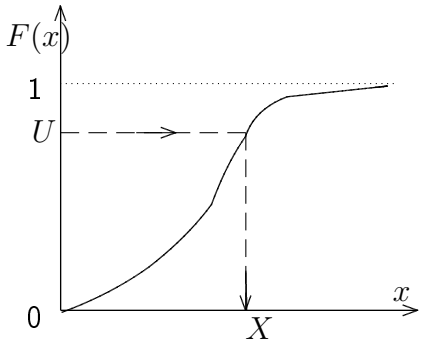

Рис. 2.1. Метод обратной функции

Для *численного* обращения применяются аппроксимации обратных функций распределения и метод половинного деления. Решение уравнений вида (2.4.1) для получения каждого нового числа X (а их нужны десятки и сотни тысяч) требует больших затрат машинного времени, в особенности если аналитическую формулу получить не удается. Поэтому широко применяются приближенные методы. Наиболее простой из них состоит в кусочно-линейной аппроксимации обратной функции распределения. Он широко использовался в системе моделирования GPSS.

Неудобством табличных методов является необходимость построения новых таблиц для всех комбинаций параметров моделируемого распределения даже при сохранении его типа. Исключениями здесь являются показательное и нормальное распределения, для которых переход от стандартного к произвольному распределению производится линейным преобразованием. Это оправдывает применение для названных распределений более сложных аппроксимаций.

Недостатки применения аппроксимаций:

- метод неприменим, если F часто меняется;
- если ФР заменяется отношением полиномов или применяется кусочно-полиномиальная аппроксимация, надо хранить много коэффициентов;
- точность аппроксимации фиксирована, и при необходимости ее повышения надо менять всю функцию;
- возникают трудности при аппроксимации многопараметрических семейств.

Основные затраты на реализацию линейных аппроксимаций связаны с поиском интервала опорных значений, в который попадает очередной аргумент U. Трудоемкость последовательного просмотра для таблицы длины n в среднем составляет $n/2$. При двоичном поиске среднее число попыток уменьшается до $[\log_2 n] + 1$.

2.4.2. Равномерное и треугольное распределения

На первый взгляд простейшим является распределение, равномерное на отрезке $[a - l, a + l]$. Во всяком случае, именно оно обычно постулируется при недостатке реальных данных и фигурирует в большей части примеров из руководств к популярной системе имитационного моделирования GPSS. Плотность этого распределения равна $1/(2l)$ внутри упомянутого интервала и нулю — вне его. Начальные моменты равномерного распределения

$$f_k = \frac{(a+l)^{k+1} - (a-l)^{k+1}}{2l(k+1)}.$$

Его дисперсия $D = l^2/3$. Числа, равномерно распределенные на $[a - l, a + l]$, генерируются по правилу

$$X = a + 2 * l * U.$$

Можно показать, что сумма двух равномерно распределенных величин имеет *треугольное* распределение (Симпсона), границы которого равны суммам левых и правых границ слагаемых соответственно. В симметричном случае для отрезка $[a - l, a + l]$ максимальное значение плотности в точке a равно $1/l$.

2.4.3. Распределение Парето

Распределение Парето (оно же гиперболическое, или степенное) имеет функцию распределения

$$F(t) = 1 - (K/t)^{\alpha}, \quad t \geq K. \tag{2.4.2}$$

Момент распределения Парето порядка m

$$f_m = \alpha K^{\alpha} \cdot \frac{t^{m-\alpha}}{m-\alpha}\Bigg|_K^{\infty}$$

имеет конечное значение лишь при $\alpha > m$.

Для выработки псевдослучайных чисел, подчиненных распределению Парето, воспользуемся методом обратной функции. Из условия

$$U = (K/T)^{\alpha}$$

получаем

$$T = K/U^{1/\alpha}. \tag{2.4.3}$$

При имитации систем с Парето-распределениями, особенно для значений параметра $\alpha < 2$, обязателен контрольный вывод доли заявок, получивших отказ из-за переполнения очереди. При большой очереди (а ее длина может достигать *сотен тысяч*) отказ в приеме заявки исключает из обработки случаи чрезвычайно длительного ожидания, и даже при малой вероятности отказа оценки моментов распределения ожидания окажутся сильно заниженными.

Остальные используемые в данной книге распределения определены на полуоси $[0, \infty)$. Это обстоятельство в дальнейшем дополнительно не оговаривается.

2.4.4. Распределения экспоненциальной группы

Показательное распределение имеет ФР $F(x) = 1 - e^{-\lambda x}$, и (2.4.1) превращается в $1 - e^{-\lambda X} = U$, откуда $X = -\ln(1-U)/\lambda$. Поскольку $(1-U)$ имеет то же распределение, что и U, удобнее[1] применять формулу

$$X = -\ln U/\lambda. \tag{2.4.4}$$

[1] Но с риском обращения к логарифмической функции с нулевым аргументом.

Натуральный логарифм правильной дроби отрицателен, так что расчет дает положительную величину. Сразу же предупредим авторов программ имитационного моделирования о часто встречающейся при работе с этой формулой ошибке — потере минуса. Ее следствием является *обратный ход системного времени*, катастрофически искажающий логику модели.

Обычные стандартные программы предусматривают вычисление $\ln U$ с высокой точностью, как правило излишней при имитации случайных воздействий. Поэтому целесообразно использовать более простые и быстрые вычислительные схемы. Представляя, например, случайное число U в нормализованном виде как $U = m \cdot 2^p$, можно аппроксимировать $\ln U$ произведением $-\ln 2(p - 2.6797 + 4.0391m - 1.3594m^2)$ с погрешностью, не превышающей по абсолютной величине 0.001.

Распределение Релея с функцией распределения

$$F(x) = 1 - e^{-x^2/2\sigma^2} \tag{2.4.5}$$

приводит к генератору вида

$$X = \sigma\sqrt{-2\ln U}. \tag{2.4.6}$$

Распределение Вейбулла имеет функцию распределения

$$F(x) = 1 - \exp(-x^k/T). \tag{2.4.7}$$

Соответственно оказывается, что

$$X = \sqrt[k]{-T\ln(1-U)}. \tag{2.4.8}$$

Логистическое распределение общего вида задается ФР

$$F(x) = \frac{1}{1 + e^{-(x-a)/b}}.$$

Отсюда имеем уравнение $U[1 + e^{-(X-a)/b}] = 1$ с решением

$$X = a - b\ln(1/U - 1).$$

2.5. Частные методы

Здесь мы рассмотрим ситуации, когда для моделирования случайной величины непосредственно воспроизводится вероятностная схема, порождающая интересующее нас распределение.

2.5.1. Фазовые распределения

Многие сложные СМО могут быть рассчитаны численно после аппроксимации исходных распределений «фазовыми» с показательно распределенными задержками в фазах. Поэтому моделирование с применением фазовых распределений часто необходимо для верификации аналитических результатов. Рассмотрим соответствующие способы генерации.

Распределение Эрланга r-го порядка предполагает последовательное прохождение r фаз и соответственно является распределением суммы r «показательных» чисел. Следовательно,

$$X = \sum_{k=1}^{r} \left(-\frac{1}{\lambda} \ln U_k \right) = -\frac{1}{\lambda} \sum_{k=1}^{r} \ln U_k = -\frac{1}{\lambda} \ln \prod_{k=1}^{r} U_k. \qquad (2.5.1)$$

Последний вариант уменьшает число сравнительно трудоемких операций логарифмирования.

Ниже приведена подпрограмма для моделирования эрлангова распределения длительности обслуживания:

```
 real function flib()
   real mu,x(3),p
   integer i
   data mu /5.0/
   call random_number(x)
   p=1.0
   do i=1,3; p=p*x(i); end do
   flib=-log(p)/mu
end
```

Очевидно обобщение этого подхода на случай эрлангова распределения с *различными* параметрами экспоненциальной задержки в фазах.

Гиперэрлангово распределение с возможным обходом первой фазы может быть получено в двухэтапном процессе. На первом этапе при $U > y$, где y — вероятность полнофазного обслуживания, число фаз процесса Эрланга уменьшается на единицу. Второй этап аналогичен предыдущему случаю.

Гиперэкспоненциальное распределение. Представление ДФР в виде

$$\bar{F}(t) = \sum_{i=1}^{k} y_i e^{-\mu_i t} \tag{2.5.2}$$

позволяет считать исходный процесс проходящим одну из n альтернативных фаз. Здесь $\{y_i\}, \quad 0 \le y_i \le 1, \quad \sum_{i=1}^{n} y_i = 1$, интерпретируются как вероятности выбора i-й фазы, а $\{\mu_i\}$ — как параметры показательного распределения времени пребывания в ней.

Среди названных величин могут быть комплексные (попарно сопряженные), что подчеркивает $фиктивный$ характер расщепления процесса на фазы. Допустимость таких параметров была впервые отмечена Д. Коксом в 1955 г. Однако использовать комплексные параметры при генерации случайных чисел не удается.

Гиперэкспоненциальное распределение независимо от его порядка (числа составляющих) требует $двух$ обращений к равномерному ДСЧ. С помощью числа U_1 выбирается номер экспоненты i (см. рекомендации разд. 2.4 по выбору дискретных случайных величин), а посредством U_2 формируется согласно (2.4.4) величина, имеющая показательное распределение с параметром μ_i. В нашем случае для генерации H_2-распределенных чисел необходима функция вида

```
real function flib()
  real  mu(2),x(2),p,y
  integer i
  data mu,y /.../
  call random_number(x)
  p=-log(x(1))
  if (x(2)<y) then
     p=p/mu(1)
  else
     p=p/mu(2)
  end if
  flib=p
end function
```

2.5.2. Нормальное распределение и родственные ему

Нормальное распределение описывается плотностью

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2}.$$

Известно,что распределение суммы n одинаково и независимо распределенных случайных величин со средним значением $z^{(1)}$ и дисперсией $d^{(1)}$ при $n \to \infty$ стремится к нормальному распределению с параметрами $z^{(n)} = nz^{(1)}$ и дисперсией $d^{(n)} = nd^{(1)}$. Достаточно хорошая сходимость наблюдается уже при $n > 6$. В [3, с. 145] приведены варианты генерации нормально распределенных чисел, требующие пяти и даже двух равномерных чисел.

Логарифмически нормальное распределение с плотностью

$$f(x) = \frac{\exp[-(\ln(x) - \mu)^2/2\sigma^2]}{x\sigma\sqrt{2\pi}}$$

генерируется как экспонента от нормально распределенного $N(\mu, \sigma^2)$.

2.6. Дискретные распределения — общий подход

Для формирования дискретных случайных величин непрерывная функция распределения заменяется ступенчатой кумулянтой. Метод *последовательных сравнений* является дискретным аналогом метода обратной функции. Он заключается в переборе значений X, пока не окажется

$$F(X-1) = \sum_{i<X} p_i < U \le \sum_{i \le X} p_i. \qquad (2.6.1)$$

При этом $\Pr(X = i) = F(i) - F(i-1) = p_i$. Метод неудобен для распределений с «толстыми хвостами». Возможная модификация — половинное деление при работе с таблицей функции распределения. Тот же метод можно применить для случайного розыгрыша возможных событий, если эти события предварительно перенумеровать и упорядочить по убыванию соответствующих вероятностей.

Бернуллиево распределение имеет бинарная $\{0,1\}$ величина, принимающая значение 1 с вероятностью p. Здесь X получает значение 1 при $U \le p$ и 0 — в противном случае.

Геометрическое распределение

$$P_r = p(1-p)^{r-1}$$

описывает число испытаний до первого успеха, который в каждой из независимых попыток достигается с вероятностью p. Бернуллиевы попытки выполняются до получения первого числа $U \le p$; X равен полному числу попыток.

Биномиально распределенное значение определяется вероятностями

$$P_r = \binom{N}{r} p^r (1-p)^{N-r}.$$

Это вероятность точно r успехов в N попытках. Таким образом, следует сформировать N чисел $\{U_i\}$ и подсчитать, сколько из них не превысило p.

Распределение Пуассона. На интервале $[0,t]$ происходит ровно K событий пуассоновского потока интенсивности λ, если сумма K показательно с параметром λ распределенных случайных чисел меньше t, а сумма $K+1$ таких чисел — больше t. Таким образом, K есть наибольшее целое число, для которого

$$-\frac{1}{\lambda}\sum_{k=1}^{K} \ln U_k < t$$

или — в более удобной форме —

$$\prod_{k=1}^{K} U_k > e^{-\lambda t}.$$

При известном среднем λt здесь не нужны ни логарифмирование, ни вычисление экспоненты (исключая установку датчика). Возможна работа с предварительно сгенерированным случайным временем T.

Составной пуассоновский поток генерируется с одновременным моделированием объема каждой заявки и накоплением суммарного объема. Последняя проблема характерна, например, для эшелонированных

систем управления запасами (спрос в высших звеньях формируется групповыми заявками из низших звеньев).

Распределение Пуассона за случайный интервал. Здесь моделируется число K событий с распределением

$$P\{K = k\} = q_k = \int_0^\infty \frac{(\lambda t)^k}{k\,!} e^{-\lambda t}\, dF(t).$$

Этот случай сводится к предыдущему, если предварительно сформировать случайный интервал T, подчиненный закону $F(t)$.

Контрольные вопросы

1. Сопоставьте физические и программные ДСЧ.
2. Как обеспечить непересекаемость серий псевдослучайных чисел? Запрограммируйте функцию получения псевдослучайного числа с заданным номером.
3. При решении каких задач может оказаться полезной идея ускоренного умножения? Запрограммируйте одну из этих задач.
4. Примените метод обратной функции для генерации чисел с распределением Вейбулла.
5. Как использовать распределение Релея для генерации пары нормально распределенных чисел? Каков КПД этого метода?
6. Как сгенерировать случайное число событий простейшего потока за указанный интервал?

Глава 3

Обработка результатов моделирования

3.1. Общие положения

Анализ результатов моделирования должен основываться на надежных статистических процедурах. В частности, необходимо получение доверительных интервалов, а также установление зависимости между временем (числом циклов) моделирования и точностью оценок.

Обозначим через $\hat{a}(N)$ статистическую оценку параметра a по данным N опытов. Оценивание одного и того же параметра, вообще говоря, может быть проведено различными способами. Наилучшими («подходящими») считаются оценки, удовлетворяющие требованиям состоятельности, несмещенности и эффективности.

Оценка называется $состоятельной$, если она при неограниченном увеличении числа опытов $сходится\ по\ вероятности$ к искомому значению параметра:

$$\lim_{N\to\infty} P\{|\hat{a}(N) - a| < \varepsilon\} = 1.$$

Оценка является $несмещенной$, если ее математическое ожидание при любом $конечном\ N$ равно истинному значению параметра:

$$M[\hat{a}(N)] = a.$$

Оценка $эффективна$, если среди всех возможных она обладает наименьшей дисперсией.

Один и тот же параметр часто можно оценить разными способами. Пусть даны упорядоченные по величине результаты измерений $\{x_1, \ldots, x_n\}$ нормально распределенной случайной величины (n — нечетное). Требуется оценить ее среднее значение a. В качестве статистической оценки параметра a можно выбрать:

1) среднее арифметическое $\hat{a}_1 = \sum_i x_i/n$;

2) выборочную медиану $\hat{a}_2 = x_{[n/2]+1}$;

3) выборочную середину размаха $\hat{a}_3 = (x_1 + x_n)/2$.

Любая из трех этих статистик дает несмещенную оценку среднего значения a, но по точности эти оценки существенно различаются — с *философской* точки зрения уже потому, что лишь первая из них использует *всю* информацию, содержащуюся в выборке. Если $\{x_i\}$ распределены по нормальному закону с неизвестным средним и единичной дисперсией, то

$$D[\hat{a}_1] = 1/n; \qquad D[\hat{a}_2] = 1.57/n; \qquad D[\hat{a}_3] = \pi/24 \lg n.$$

Для получения оценки $\hat{a}$ со среднеквадратическим отклонением, равным 0.01, в первом случае требуется 100 измерений, во втором — 157, а в третьем — $6.6 \cdot 10^{17}$.

3.2. Точечные оценки

3.2.1. Оценивание вероятностей

Оценкой вероятности состояния r служит частота его появления

$$\hat{p}_r = N_r/N, \tag{3.2.1}$$

где N_r — количество реализаций (из общего числа N), в которых наблюдалось интересующее нас событие. В моделях с непрерывным временем общее число реализаций N заменяется на системное время T протекания процесса, а N_r — на суммарное время T_r наблюдения состояния r в ходе моделирования:

$$\hat{p}_r{}' = T_r/T. \tag{3.2.2}$$

Очевидно, что усреднение по системному времени дает оценки с меньшей дисперсией, чем усреднение по числу наблюдений.

3.2.2. Оценивание моментов

Несмещенной оценкой математического ожидания случайной величины X является среднее арифметическое результатов N опытов:

$$\hat{x} = \frac{1}{N}\sum_{i=1}^{N} X_i. \tag{3.2.3}$$

Аналогичными формулами выражаются начальные статистические моменты более высокого порядка:

$$\hat{x}_k = \frac{1}{N}\sum_{i=1}^{N} X_i^k, \qquad k = 1, 2, \ldots \tag{3.2.4}$$

При расчете $центральных$ моментов приходится вводить поправку, вызванную заменой истинного математического ожидания x случайной величины X в формулах вида

$$\hat{\mu}_k = \frac{1}{N}\sum_{i=1}^{N}(X_i - x)^k, \qquad k = 1, 2, \ldots,$$

статистическим средним $\hat{x}$. В частности, несмещенная оценка для статистической дисперсии

$$\hat{D} = \frac{1}{N-1}\sum_{i=1}^{N}(X_i - \hat{x})^2. \tag{3.2.5}$$

Статистическое среднеквадратическое отклонение

$$\hat{s} = \sqrt{\hat{D}}. \tag{3.2.6}$$

Эмпирический контроль сходимости оценок моментов с ростом числа испытаний позволяет убедиться в $существовании$ моментов (у распределения Коши бесконечны все моменты, у распределения Парето — начиная с некоторого номера).

Выборочные средние из-за влияния начальных условий будут, вообще говоря, $смещенными$ оценками истинного значения.

Все перечисленные оценки $состоятельны$ — следовательно, теоретически при существовании стационарного режима и неограниченном возрастании числа наблюдений N (длительности процесса T) сходятся по вероятности к искомым параметрам. Однако монотонность изменения оценок по числу испытаний не гарантируется.

3.2.3. Работа с распределением Парето

Приведем таблицы 3.1–3.2 результатов имитационного моделирования одноканальных СМО для распределений E_4, M и гиперэкспоненциального H_2 с коэффициентом вариации 2. Коэффициент загрузки принимался равным 0.8, средний интервал между заявками $a_1 = 1$. Последнее допущение позволяет на основании формулы Литтла рассматривать средние времена ожидания и как средние длины очередей. В таблице 3.2 опущены результаты для марковского обслуживания, которые, как и следовало ожидать, полностью совпадают с результатами предыдущей таблицы при простейшем входящем потоке.

Таблица 3.1. Имитационное моделирование системы GI/M/1

Тыс.	E_4			M			H_2		
испыт.	w_1	w_2	w_3	w_1	w_2	w_3	w_1	w_2	w_3
50	1.755	8.845	65.82	3.249	25.57	283.6	8.557	161.1	4321
100	1.791	9.145	68.85	3.314	26.95	313.4	8.259	146.5	3718
200	1.821	9.573	75.30	3.267	26.14	301.1	8.483	153.4	3940
500	1.818	9.482	73.49	3.178	24.86	282.3	8.771	165.9	4508
1000	1.813	9.434	72.82	3.238	25.92	301.6	8.554	156.2	4095
2000	1.805	9.379	72.94	3.220	25.52	293.4	8.472	151.4	3864
5000	1.817	9.516	74.67	3.202	25.13	285.4	8.524	154.2	3994
10000	1.817	9.518	74.52	3.183	24.81	279.8	8.536	154.6	4012
Точные	2.091	10.93	85.73	3.200	25.60	307.2	7.651	146.3	4198

Таблица 3.2. Имитационное моделирование системы M/G/1

Число	E_4			H_2		
испыт.	w_1	w_2	w_3	w_1	w_2	w_3
50	2.030	9.68	67.07	7.701	156.3	4587
100	2.079	10.57	79.61	8.086	169.6	5133
200	2.022	9.80	69.32	8.354	185.1	6105
500	2.021	9.78	69.73	8.082	167.1	5049
1000	2.020	9.79	70.50	8.052	164.5	4899
2000	2.014	9.73	69.71	8.127	168.1	5054
5000	2.001	9.59	68.40	8.058	166.8	5062
10000	1.997	9.55	67.81	8.008	163.4	4859
Точные	2.000	9.60	68.88	8.000	166.4	5197

В таблицах 3.3 и 3.4 представлены моменты распределения времени ожидания для систем Pa/M/1 и M/Pa/1 по итогам имитационных экспериментов (с тильдой указаны распределения, которые выравниваются распределением Парето с сохранением двух моментов). В последних трех колонках представлены результаты для Парето-распределения с параметром $\alpha = 1.5$.

Таблица 3.3. Имитационное моделирование системы Pa/M/1

Тыс.	$\sim E_4$			$\sim M$			$\sim H_2$			$\alpha = 1.5$		
исп.	w_1	w_2	w_3	w_1	w_2	w_3	w_1	w_2	w_3	w_1	w_2	w_3
50	1.60	7.38	48.9	2.06	11.5	94.0	2.58	16.9	154.6	8.47	153.0	3882
100	1.61	7.54	51.2	2.05	11.4	92.8	2.56	16.9	158.0	8.61	148.6	3501
200	1.67	8.33	62.5	2.12	12.9	122.4	2.61	17.8	177.7	8.78	154.5	3720
500	1.68	8.37	62.2	2.14	12.8	117.4	2.63	18.2	188.3	8.91	163.6	4189
1000	1.67	8.27	61.3	2.12	12.4	109.0	2.62	17.9	183.3	9.00	171.1	4693
2000	1.68	8.36	62.9	2.15	12.7	111.1	2.63	18.2	190.5	9.16	180.3	5202
5000	1.68	8.35	62.1	2.15	12.7	112.4	2.62	18.0	187.6	9.01	174.5	4960
10000	1.69	8.37	62.2	2.15	12.7	112.4	2.61	17.8	182.1	9.06	176.8	5083
20000	1.69	8.40	63.0	2.15	12.8	113.9	2.62	17.9	183.7	9.10	179.9	5324

Таблица 3.4. Имитационное моделирование системы M/Pa/1

Тыс.	$\sim E_4$			$\sim H_2$			$\alpha = 1.5$		
испыт.	w_1	w_2	w_3	w_1	w_2	w_3	w_1	w_2	w_3
10	2.12	11.9	104.8	4.81	89.1	2702	65.5	1.32e4	3.34e6
20	2.07	11.1	90.5	4.53	78.1	2215	58.1	1.11e4	2.76e6
50	1.97	9.8	77.0	4.38	101.6	5161	48.6	9.80e3	2.54e6
100	1.90	9.0	66.4	3.93	77.5	3398	32.9	6.03e3	1.50e6
200	2.01	10.3	81.3	4.22	112.1	8111	42.4	2.08e4	1.82e7
500	2.02	10.5	84.8	4.19	102.8	6194	37.2	1.38e4	9.53e6
1000	2.00	10.4	86.4	4.30	118.5	8933	45.5	2.74e4	3.05e7
2000	2.00	10.4	88.6	9.70	8644	1.59e7	106.3	3.02e6	1.63e11
5000	2.00	10.5	93.8	6.70	3626	6.41e6	70.8	1.22e6	6.53e10
10000	2.00	10.8	112.0	6.16	2178	3.38e6	81.2	7.32e5	3.36e10
20000	1.99	10.7	105.1	5.46	1272	1.76e6	68.7	4.11e5	1.71e10

Как видно из приведенных таблиц, модель Pa/M/1 ведет себя много лучше, чем M/Pa/1. Дело в том, что основные характеристики системы

GI/M/1 выражаются через корень ω уравнения

$$\omega = \int_0^\infty e^{-\mu(1-\omega)t}\, dA(t), \tag{3.2.7}$$

решаемого методом итераций для конкретных распределений $A(t)$ интервалов между смежными заявками. Для сходимости итераций должно выполняться условие $\rho < 1$, в данном случае сводящееся к $a_1\mu > 1$. При его выполнении распределение моментов виртуального ожидания в системе GI/M/1 подчинено показательному закону с параметром $\mu(1-\omega)$. Соответственно его моменты

$$w_k = \frac{\rho k!}{[\mu(1-\omega)]^k}, \quad k = 1, 2, \ldots \tag{3.2.8}$$

При $1 < \alpha < 2$ значения ω близки к 1, что и приводит к чрезвычайно быстрому росту моментов распределения ожидания.

С другой стороны, для системы M/G/1 из формулы Полячека—Хинчина следует, что для существования k-го момента ожидания необходима конечность моментов распределения обслуживания порядка $k+1$. Для рассматриваемых нами E_4, M и H_2 имеем значения $\alpha =$ 3.2360, 2.2142 и 2.1180 соответственно, так что в первом случае должны сходиться оценки двух моментов, во втором и третьем — только w_1, причем сходимость заметно ухудшается с уменьшением α. Для $\alpha = 1.5$ расходится даже оценка первого момента.

Результаты моделирования заметно различны для систем с «классическими» и с Парето-распределениями. Выскажем следующие соображения:

1. Распределение Парето, будучи двухпараметрическим, может выравнивать лишь два момента исходного распределения. Поэтому хорошего согласия с методиками, учитывающими большее число моментов, ожидать не приходится. Заметим, что в рассмотренной ситуации для распределения Парето третий момент больше, чем третий момент выравниваемых. Поэтому при использовании этого распределения для модели M/G/1 ожидание увеличивается, а для GI/M/1 — уменьшается.

2. Расхождения просто обязаны быть существенными в случае, когда выравнивающее распределение Парето имеет бесконечные моменты невысокого порядка.

3. Расхождения заметно возрастают при увеличении вариации исходного распределения.

4. В таблицах отмечается предсказываемое автором [20] «волнообразное» изменение оценок.

Моделирование многоканальных систем. Парето-обслуживание в одноканальных системах приводит к очень редким занятиям системы на чрезвычайно длительный срок, что и порождает описанные выше эффекты. Можно предполагать, что случаи одновременного занятия «супердлинными» заявками всех каналов окажутся практически невероятными. Таким образом, при фиксированном коэффициенте загрузки дробление производительности вычислительной системы должно оказаться хорошей защитой от сверхдолгого ожидания.

В таблице 3.5 для сравнения с таблицей 3.4 представлены результаты имитационного моделирования системы M/Pa/2, полученные удвоением среднего времени обслуживания.

Таблица 3.5. Имитационное моделирование системы M/Pa/2

Тыс.	$\sim E_4$			$\sim H_2$			$\alpha = 1.5$		
испыт.	w_1	w_2	w_3	w_1	w_2	w_3	w_1	w_2	w_3
10	2.29	16.2	184.3	7.23	346.2	27148	149.0	7.50e4	4.52e7
20	2.06	13.7	152.2	6.09	216.7	14712	107.2	4.57e4	2.52e7
50	1.89	10.7	101.3	4.11	112.0	6841	61.0	2.19e4	1.10e7
100	1.82	9.87	88.3	3.40	72.0	3844	36.2	1.14e4	5.56e6
200	1.82	9.60	81.1	3.33	61.1	2719	39.4	1.72e4	1.17e7
500	1.78	9.07	71.8	3.46	62.5	2468	28.9	9.34e3	5.45e6
1000	1.81	9.46	79.8	3.47	66.7	3197	32.4	1.58e4	1.51e7
2000	1.78	9.22	76.4	3.50	76.8	5076	33.2	1.57e4	1.43e7
5000	1.78	9.11	75.6	3.57	91.6	7812	39.0	2.44e4	3.04e7
10000	1.78	9.25	82.1	3.93	171.6	30784	49.3	4.56e4	8.08e7
20000	1.77	9.17	80.4	3.89	185.1	44305	52.0	4.86e4	8.59e7

Сопоставление этих таблиц показывает заметную тенденцию к уменьшению как самих моментов распределения времени ожидания, так и их изменчивости по числу испытаний.

3.3. Дисперсия и корреляция

При любом конечном N или T получаемые оценки *случайны*, в связи с чем возникает вопрос об их качестве (погрешности). Простейшим способом определения качества оценки является указание ее *дисперсии*. Известно [17], что дисперсия оценки $\hat{\nu}_r$ r-го начального момента

$$D[\hat{\nu}_r] = (\nu_{2r} - \nu_r^2)/N \tag{3.3.1}$$

(в правую часть входят теоретические значения соответствующих моментов). В частности, дисперсия статистического среднего

$$D[\hat{\nu}_1] = D_X/N. \tag{3.3.2}$$

Воспользуемся этой формулой для вычисления дисперсии оценки $\hat{v}$ среднего времени пребывания заявки в системе $M/M/1$, которое имеет показательное распределение с параметром $(\mu - \lambda)$. Тогда

$$D[\hat{v}] = \frac{1}{N}(\mu - \lambda)^{-2} = \frac{1}{N\mu^2}\frac{1}{(1-\rho)^2},$$

где

- λ — интенсивность входящего потока,
- μ — интенсивность обслуживания,
- ρ — коэффициент загрузки системы.

Можно ожидать качественно той же зависимости этой дисперсии от ρ и N для многоканальных систем и произвольных распределений. Кратность увеличения дисперсии времени ожидания при коэффициентах загрузки $\rho =$ 0.7, 0.9, 0.95, 0.99 по отношению к случаю $\rho = 0.5$ равна соответственно 3, 25, 100 и 2500. Это объясняет общеизвестную трудность моделирования СМО при большой загрузке ($\rho \to 1$).

Знание дисперсий используется для сопоставления альтернативных оценок одного параметра. Как уже отмечалось, усреднение по системному времени дает оценки с меньшей дисперсией, чем усреднение по числу наблюдений.

3.4. Метод квантилей

Альтернативой определения параметров распределений через статистические оценки моментов является *метод квантилей* — выравнивание значений статистической функции распределения или ДФР для заданных аргументов. Его применение не связано со сходимостью оценок для высших моментов.

Из равенства (2.4.2) следует, что параметр α распределения Парето можно выразить через отношение дополнительных функций распределения:

$$\alpha = \frac{\ln(\bar{F}_1/\bar{F}_2)}{\ln(t_2/t_1)}. \tag{3.4.1}$$

Имея α, второй параметр можно подобрать по заданному первому моменту или через один из квантилей. Назначаемые квантили должны быть не слишком близкими друг к другу и допускать значительную вероятность их превышения. В таблице 3.6 приведены полученные в ходе статистического эксперимента оценки параметра α распределения Парето.

Таблица 3.6. Оценки параметра α методом квантилей

Тыс. испыт.	$\alpha = 1.5$	$\alpha = 2.5$	$\alpha = 3.5$
2	1.56133	2.50043	3.49966
5	1.51164	2.50043	3.49966
10	1.49327	2.50043	3.49966
20	1.50161	2.50043	3.49966
50	1.50560	2.50043	3.49965
100	1.49750	2.50042	3.49966
200	1.50504	2.50043	3.49965
500	1.50338	2.50039	3.49963
1000	1.50415	2.50037	3.49964
2000	1.50082	2.50034	3.49968
5000	1.50007	2.50028	3.49970
10000	1.50066	2.50016	3.49978
20000	1.50019	2.50002	3.49982
50000	1.50048	2.49989	3.49974
100000	1.50026	2.49975	3.49976

Квантили $\{t_i\}$ назначались по априорно известным данным — как $t = K/p^{1/\alpha}$ для вероятностей 1/3 и 2/3, оценка $\hat{\alpha}$ формировалась по формуле (3.4.1). Оценка второго параметра $\hat{K}$ может быть получена через $\hat{\alpha}$ и статистическое среднее.

Из данной таблицы следует, что метод квантилей весьма эффективен для определения параметра α — даже в общепризнанно опасной зоне $\alpha < 2$, где второго момента не существует.

3.5. Классический подход к интервальным оценкам

Имея оценку и ее дисперсию, можно перейти к построению *доверительного интервала*. Практические требования к оценкам формулируются в терминах показателей их точности и надежности. Под *точностью* понимается половина δ длины доверительного интервала, а под *надежностью* P — вероятность того, что истинное значение параметра окажется принадлежащим упомянутому интервалу (доверительная вероятность). При фиксированных условиях увеличение требований к точности уменьшает доверительную вероятность, а увеличение доверительной вероятности снижает точность оценок. Обычно ставится задача определения числа испытаний N, при котором будут обеспечены заданные δ и P.

Пусть определяется среднее время w ожидания начала обслуживания, причем дисперсия времени ожидания равна σ_W^2. При характерной для имитационного моделирования кратности наблюдений N разность $\hat{w} - w$ можно на основании теоремы А.М. Ляпунова считать распределенной нормально с дисперсией σ_W^2/N — см. формулу (3.3.2). Следовательно, доверительная вероятность

$$P\{|\hat{w} - w| \le \delta\} = \Phi\left(\frac{\delta\sqrt{N}}{\sigma_W\sqrt{2}}\right), \tag{3.5.1}$$

где $\Phi(\cdot)$ — функция Лапласа. Переходя к обратной функции, имеем

$$\Phi^{-1}(P) = \frac{\delta\sqrt{N}}{\sigma_W\sqrt{2}},$$

откуда требуемое число наблюдений

$$N \ge 2[\Phi^{-1}(P)]^2\,(\sigma_W/\delta)^2 = k(P)(\sigma_W/\delta)^2. \tag{3.5.2}$$

Коэффициент $k(P)$ выбирается из таблицы 3.7.

Таблица 3.7. Коэффициенты $k(P)$ для расчета числа испытаний

P	0.900	0.950	0.970	0.980	0.990	0.995	0.999
$k(P)$	2.69	3.84	4.71	5.43	6.66	7.90	10.82

Таким образом, необходимое число испытаний обратно пропорционально *квадрату* допустимой относительной погрешности и заметно возрастает с повышением доверительной вероятности.

В случае малого количества наблюдений (порядка нескольких десятков) доверительные интервалы для нормально распределенных величин строят с помощью *распределения Стьюдента*.

При оценке *вероятностей* в формулу (3.5.2) вместо σ следует подставить $\sqrt{p(1-p)}$. Тогда формула (3.5.2) примет вид

$$N \geq k(P)\frac{p(1-p)}{\delta^2}. \tag{3.5.3}$$

При определении вероятностей обычно задаются их *относительной* точностью, то есть считают $\delta = \varepsilon p$. Тогда для малых вероятностей $1-p \approx 1$, и можно переписать (3.5.3) в виде

$$N \geq \frac{k(P)}{p\varepsilon^2}. \tag{3.5.4}$$

Итак, необходимое число испытаний обратно пропорционально искомой вероятности и *квадрату* допустимой относительной погрешности ее. Например, для определения вероятности порядка 10^{-6} с относительной погрешностью в 1% и доверительной вероятностью 0.98 должно быть проведено $N > 5.43 \cdot 10^6 \cdot 10^4 \approx 5 \cdot 10^{10}$ испытаний. Это существенно затрудняет *непосредственную* оценку вероятностей редких событий методом имитационного моделирования.

3.6. Регенерирующий процесс и его обработка

Для применения описанных выше классических процедур построения доверительных интервалов выходные данные должны образовывать набор статистически независимых и одинаково распределенных

выборочных значений. Мы уже отмечали, что при большом времени W_k ожидания k-го требования W_{k+1} также будет большим с высокой вероятностью. Таким образом, значения W_k и W_{k+1} будут сильно коррелированы, причем независимо от начальных условий. Приведем (таблица 3.8) соответствующие экспериментальные данные, полученные автором для модели $M/D/2$.

Таблица 3.8. Корреляция последовательных времен ожидания

Коэффициент загрузки	0.3	0.5	0.7	0.9
Среднее время ожидания	0.113	0.341	0.917	3.170
Коэффициент корреляции	0.390	0.597	0.807	0.951

Указанное затруднение можно преодолеть, если работать с *регенерирующими* (регулярно восстанавливающими свои свойства) процессами. Разобьем период моделирования на *циклы*, состоящие из смежных периодов занятости и простоя. Началом цикла будем считать момент прихода заявки в свободную систему. Поскольку каждый цикл начинается при одних и тех же условиях, результаты последовательных циклов статистически независимы и имеют одинаковые распределения. Пусть всего проведено N циклов. Обозначим

ν_i — число заявок, обслуженных в i-м цикле,
W_i — суммарная длительность ожидания в нем,
$n = \sum_{i=1}^{N} \nu_i$ — общее число обслуженных заявок.

Тогда оценка среднего времени ожидания

$$\hat{w} = \frac{W_1 + W_2 + \ldots W_N}{n} = \frac{(W_1 + W_2 + \ldots W_N)/N}{(\nu_1 + \nu_2 + \ldots \nu_N)/N}. \qquad (3.6.1)$$

При $N \to \infty$ в силу закона больших чисел числитель и знаменатель стремятся к своим математическим ожиданиям, так что

$$M[\hat{w}] = M[W]/M[\nu]. \qquad (3.6.2)$$

Описанный подход может быть перенесен на оценку других характеристик случайного процесса. Пусть дана последовательность $\{X_i\}$ случайных векторов размерности k и $f(x)$ — действительная функция

компонент случайного вектора. Тогда, определив Y_j как сумму значений f на j-м цикле регенерации

$$Y_j = \sum_{i=\tau_j}^{\tau_{j+1}-1} f(X_i),$$

где $\{\tau_j\}$ — номера наблюдений, открывающие j-й цикл, можно вновь оценить математическое ожидание $f(X)$ по формуле типа (3.6.1).

Для процессов с непрерывным временем считают

$$Y_j = \int_{\tau_j}^{\tau_{j+1}} f(X(t))\,dt,$$

где $\{\tau_j\}$ — уже не номера, а сами $точки$ регенерации, а $\{\nu_j\}$ из формулы (3.6.1) суть продолжительности соответствующих циклов.

В книгах [18, 45] описана процедура построения 100(1- δ)–процентного доверительного интервала для $r = M[f(x)]$ при числе циклов N порядка сотен и более.

Решающими для применения описанной методики являются следующие требования:

- процесс неоднократно возвращается в некоторое фиксированное состояние (подмножество состояний);
- среднее время между возвращениями конечно;
- момент очередного возвращения является точкой регенерации.

Получение на базе данной теории достаточно надежных оценок требует увеличения длительности моделирования — в особенности при большой загрузке, когда случаи освобождения системы становятся редкими. В связи с этим возникает проблема построения процессов, которые $приближенно$ могут рассматриваться как регенерирующие, но имеют умеренную длительность циклов.

Примером приближенного подхода к построению регенерирующего процесса может служить метод $меченых\ заявок$ при исследовании $сетей$ обслуживания, в которых неизбежно взаимное влияние заявок на возникающие задержки. Если в каждый момент времени в сети находится только одна меченая заявка, а сеть достаточно сложна, то времена

пребывания в сети последовательных меченых заявок окажутся практически некоррелированными. Этот выигрыш в «чистоте эксперимента» приходится оплачивать: кратность замедления набора статистики равна среднему числу заявок, находящихся в сети.

В заключение отметим, что ключевым моментом регенеративного подхода является использование отношений сумм случайных величин. Но равенство

$$M\Big[\sum_{i=1}^{N} W_i \Big/ \sum_{i=1}^{N} \nu_i\Big] = M\Big[\sum_{i=1}^{N} W_i/N\Big] \Big/ M\Big[\sum_{i=1}^{N} \nu_i/N\Big]$$

верно лишь асимптотически — при $N \to \infty$. Таким образом, получаемые оценки оказываются *смещенными*.

Просуммируем недостатки регенеративного метода:

- смещенность (нулевая асимптотически) получаемых оценок;
- априорная неизвестность длины цикла, что затрудняет планирование прогонов;
- большая длительность циклов, в особенности для систем с коэффициентом загрузки, близким к единице;
- трудность синхронизации параллельных прогонов при использовании методов уменьшения дисперсии результатов моделирования.

С другой стороны, его несомненными достоинствами являются:

- независимость наблюдений в различных циклах, позволяющая проводить корректную статистическую обработку результатов;
- отсутствие проблемы начальных условий.

Контрольные вопросы

1. Какими свойствами должны обладать статистические оценки? Дайте формальное определение этих свойств.
2. Приведите формулы для точечных статистических оценок среднего и дисперсии.
3. Получите рекуррентную по числу наблюдений формулу для оценки среднего.
4. Чем характеризуются точность и надежность статистических оценок? Дайте определение доверительного интервала и доверительной вероятности. Как связаны эти понятия при фиксированном числе испытаний?
5. Что вы знаете о распределении Парето? Чем оно примечательно?
6. Укажите особенности имитационного моделирования систем с распределением Парето.
7. Как требуемое число испытаний связано с определяемой малой вероятностью и относительной погрешностью оценки?
8. Дайте характеристику метода регенерирующих процессов.

Глава 4

Понижение дисперсии

4.1. Проблема и методы

Для получения оценок с достаточно узкими доверительными границами в рамках обсуждавшихся выше классических подходов необходимо очень большое число наблюдений, сильно зависящее от дисперсии результатов. В условиях фантастического роста быстродействия ЭВМ этот недостаток перестал быть критически важным. Однако остаются задачи, в которых проблема понижения дисперсии результатов моделирования остается актуальной:

- анализ сильно загруженных систем массового обслуживания;
- расчет структурно сложных систем с критически зависящими от состояния выходными показателями (например, те же СМО с отказами каналов);
- расчет вероятностей редких событий (здесь требуемое число испытаний для получения оценки с заданной относительной точностью обратно пропорционально искомой вероятности и квадрату допустимой относительной погрешности).

Этот недостаток метода был осознан и отчасти преодолен уже при первых попытках моделирования на ЭЦВМ. Еще фон Нейман и Улам в рамках Манхэттенского проекта при моделировании рассеивания нейтронов отказались от принципа «грубой силы» и применяли методы понижения дисперсии (МПД) результатов *единичного наблюдения*. Более развитые формы МПД предложены и описаны достаточно давно — см.

[13, 14, 27, 30], но до сих пор известны лишь узкому кругу теоретиков. К МПД-методам относят:

- выбор оценок с наименьшей дисперсией,
- контрольные переменные,
- дополняющие выборки,
- параллельные выборки,
- условную имитацию.

При окончательном суждении о целесообразности того или иного метода необходимо учитывать не только выигрыш от уменьшения дисперсии, но и накладные расходы вследствие усложнения процесса моделирования и обработки результатов, а также необходимости дополнительного программирования. Заметим, что их общей идейной основой является *использование дополнительных знаний* о моделируемой системе. Технологии, не привлекающие таких знаний, могут «отметаться с порога» — по аналогии с проектами «вечных двигателей».

4.2. Выбор оценок с наименьшей дисперсией

Прежде всего напомним, что усреднение по модельному времени дает оценки с меньшей дисперсией, чем усреднение по числу наблюдений. В частности, стационарную вероятность иметь в системе k заявок лучше оценивать по формуле

$$\hat{p}_k = \sum_i t_i(k)/T, \tag{4.2.1}$$

чем согласно

$$\hat{p}_k{}' = N(k)/N. \tag{4.2.2}$$

Здесь

$t_i(k)$	—	длина i-го отрезка времени, проведенного системой в k-м состоянии,
T	—	системное время моделирования,
$N(k)$	—	число наблюдений системы в k-м состоянии,
N	—	полное число наблюдений.

Разумеется, для возможности сопоставления оценок $\hat{p}_k$ и $\hat{p}_k'$ необходима представимость системы вложенной цепью Маркова. Из теоремы PASTA (Poisson Arrival See Time Average) следует, что распределение числа заявок перед прибытием заявки в СМО с пуассоновым входящим потоком совпадает с аналогичным стационарным распределением. С другой стороны, из соображений симметрии следует совпадение распределений числа заявок непосредственно перед прибытием и сразу после завершения обслуживания.

Еще один часто обсуждаемый способ этого класса — определение среднего времени пребывания заявки в системе через статистическое среднее времени ожидания и известное априорно среднее время обслуживания. Оценка среднего времени пребывания заявки в системе

$$\hat{v} = \hat{w} + \hat{b}, \tag{4.2.3}$$

где

$\hat{w}$ — оценка среднего времени ожидания обслуживания,

$\hat{b}$ — оценка средней длительности обслуживания.

Дисперсия нашей оценки

$$D[\hat{v}] = D[\hat{w}] + D[\hat{b}], \tag{4.2.4}$$

поскольку времена ожидания и обслуживания статистически независимы. Заменим оценку $\hat{b}$ в (4.2.3) априорно известным математическим ожиданием ее:

$$\bar{v} = \hat{w} + b. \tag{4.2.5}$$

Дисперсия постоянной величины b равна нулю[1]; следовательно, оценка (4.2.5) будет иметь меньшую дисперсию, чем (4.2.3). Выигрыш в точности будет расти с уменьшением загрузки системы по двум причинам:

- при малой загрузке время пребывания в основном определяется чистой длительностью обслуживания;
- уменьшение загрузки уменьшает дисперсию времени ожидания.

Заметный эффект возможен, если дисперсия чистого времени обслуживания соизмерима с дисперсией времени ожидания. Итак, для определения средних характеристик СМО посредством имитационного моделирования целесообразно оценить среднее время ожидания $\hat{w}$, после чего

[1] Это не значит, что длительность обслуживания постоянна!

вычислить среднее время пребывания и (по Литтлу) среднее число заявок в системе. Разумеется, эти рекомендации не исключают независимого получения всех упомянутых характеристик в процессе моделирования — для контроля правильности работы модели посредством законов сохранения.

В заключение отметим, что расчет средней длины очереди через среднее время ожидания заявки требует детального моделирования динамики очереди: запоминания моментов прибытия заявок, их продвижения и т. п. При прямом расчете очереди во всем этом нет необходимости. Теперь ясно, чем оплачивается повышение точности оценок через временные показатели.

Аналогичный подход можно применить к вычислению длины цикла занятости T, состоящего из периода занятости T_B и периода простоя T_I. Здесь T_B следует оценить на модели, а длительность простоя определить по формуле для среднего остатка интервала между заявками

$$M[T_I] = a_2/(2a_1),$$

где $\{a_i\}, \quad i = 1, 2$ — начальные моменты распределения интервалов между смежными заявками рекуррентного потока. Для простейшего потока интенсивности λ

$$M[T_I] = a_1 = 1/\lambda.$$

4.3. «Противоположные» выборки

Проведем два *зависимых* опыта и возьмем в качестве оценки искомой величины полусумму их результатов

$$\tilde{X} = (X_1 + X_2)/2. \tag{4.3.1}$$

Дисперсия этой оценки

$$D[\tilde{X}] = (D[X_1] + D[X_2] + 2K_{X_1X_2})/4. \tag{4.3.2}$$

Если принять $D[X_1] = D[X_2]$, а этого легко добиться соответствующей организацией эксперимента, то

$$D[\tilde{X}] = D_X(1 + \xi)/2, \tag{4.3.3}$$

где ξ — коэффициент корреляции X_1 и X_2. При *отрицательном* ξ дисперсия усредненного результата двух опытов может быть существенно уменьшена и в идеальном случае ($\xi = -1$) сведена к нулю. Аналог описанного приема широко используется в измерительной технике в виде разнообразных *компенсационных схем*.

Обычный способ получения отрицательной корреляции — брать в одном опыте исходное случайное число из [0,1), а в другом его дополнение до единицы (отсюда термин `antithetic variables` — противоположные переменные). Для этого в датчик псевдослучайных чисел можно встроить вспомогательную переменную — «кувыркающуюся единицу», знак которой определяет, какое из упомянутых действий должно быть выполнено при текущем обращении:

```
.....
u=random()
if (c<0) then
   u=1.0-u
end if
dt =-log(u)/lam ! для EXP закона
c=-c
```

Эта переменная должна сохранять свое значение между обращениями к подпрограмме, для чего в Алголе 60 использовался описатель own, в ПЛ/1 — `STATIC`. В Фортране 90 это свойство обеспечено автоматически.

Для *пары прогонов* можно использовать дополнение начальной установки до модуля датчика — получаемые числа $\{U_i\}$ (проверено!) будут взаимно дополняющими.

Следует иметь в виду, что последующие нелинейные функциональные преобразования могут «смягчить» запланированную противоположность. В [31, с. 89–90] показано, что коэффициент корреляции «дополняющих» переменных для равномерного и треугольного распределений остается равным -1. По-видимому, так же обстоит дело для всех распределений с симметричной плотностью.

Моделирование дало -0.6445 для показательного закона, -0.9475 для закона Релея и практически -1 — для треугольного закона, что и предсказывалось. По расчету для показательного закона дисперсия должна уменьшиться втрое, а для распределения Релея — почти в 20

раз. Заметим, что плотность последнего *визуально* очень далека от симметричной.

В книге [13] обсуждается также способ преодоления упомянутой нелинейности с помощью правила

$$X_1 = 2\bar{x} - X_1.$$

Оно гарантирует равенство $\xi_{X_1X_2} = -1$.

Ключевыми элементами описанной технологии являются монотонность преобразования случайных величин от входа к выходу (это обязательное условие отрицательной корреляции результатов «противоположных» опытов) и *синхронизация* наблюдений. Для последней необходимы:

- отдельный ДСЧ на каждую генерируемую случайную величину: интервал между заявками, время обслуживания, номер следующего узла сети и т. п. (вопрос о начальных установках нами уже обсуждался);
- возможность либо фиксировать числа $\{U_i\}$ одного прогона для их использования в параллельном, либо запоминать начальную установку соответствующих ДСЧ и затем повторно генерировать ту же серию.

4.4. Контрольные переменные

Здесь мы ограничимся расчетной схемой для одномерного случая[2]. Пусть X — случайная величина с неизвестным математическим ожиданием $M[X] = x$ и $\hat{x}$ — его несмещенная оценка, найденная стандартными методами главы 3. Случайная переменная Y может быть выбрана в качестве *контрольной переменной* для X, если она коррелирована с X и имеет известное математическое ожидание y.

Попытаемся с помощью контрольной переменной получить несмещенную оценку x с дисперсией меньшей, чем у $\hat{x}$. Введем новую случайную величину $X(c) = X - c(Y - y)$. Для любой константы c выражение

$$\bar{x}(c) = \hat{x} - M[c(Y - y)] \tag{4.4.1}$$

[2] Многомерный вариант см. в [27, 30].

также даст несмещенную оценку x, поскольку $M[Y - y] = 0$. Дисперсия контролируемой случайной величины

$$D[X(c)] = D[X - cY] = D[X] - 2cK_{XY} + c^2 D[Y], \tag{4.4.2}$$

если $c > 0$ и корреляционный момент $K_{XY} > cD[Y]/2$, будет меньше $D[X]$. Приравнивая нулю производную правой части (4.4.2) по c, получаем оптимальное значение поправочного коэффициента

$$c^* = K_{XY}/D[Y]. \tag{4.4.3}$$

Минимальная дисперсия $D[X(c)]$ достигается при подстановке c^* и равна

$$\begin{aligned} D^* &= D[X] - 2K_{XY}^2/D[Y] + K_{XY}^2/D[Y] = D[X] - K_{XY}^2/D[Y] \\ &= D[X](1 - K_{XY}^2/(D[X]D[Y])) = D[X](1 - \xi_{XY}^2). \end{aligned} \tag{4.4.4}$$

Ее величина стремится к нулю по мере приближения к единице модуля коэффициента корреляции ξ_{XY} — независимо от знака последнего.

Описанный метод является одним из самых элегантных и многообещающих МПД. Он имеет глубокий естественно-научный смысл — *привлечение дополнительных теоретических знаний* об исследуемом процессе, что, собственно, и позволяет уменьшить неопределенность в результатах моделирования. Проблема состоит в надлежащем подборе контрольной переменной.

4.5. Параллельные выборки

Часто приходится сравнивать разные варианты моделируемых систем, различающиеся параметрически или даже структурно. В таких случаях важны не столько абсолютные значения показателей работы сравниваемых систем, сколько их разность, которую желательно оценить возможно точнее. Результатом эксперимента будет разность показателей двух туров

$$\tilde{X} = X_1 - X_2. \tag{4.5.1}$$

Ее дисперсия

$$D[\tilde{X}] = D[\tilde{X}_1] + D[\tilde{X}_2] - 2K_{X_1X_2}$$

в случае близких дисперсий составляющих равна

$$D[\tilde{X}] \approx 2D[X] \cdot (1 - \xi). \tag{4.5.2}$$

Отсюда вытекает очевидная практическая рекомендация: использовать в параллельных турах одни и те же серии случайных чисел $\{U_i\}$ (разумеется, каждую по своему назначению).

Поскольку данный метод рекомендуется для сравнения $различных$ систем, проблема синхронизации серий случайных чисел здесь может осложниться.

4.6. Условная имитация

Условная имитация, как и метод контрольных переменных, комбинирует имитацию и дополнительные знания о моделируемом процессе, но является гораздо более гибким инструментом понижения дисперсии.

4.6.1. Расслоенные выборки

Идея использования расслоенных (стратифицированных) выборок восходит к демографической статистике и методикам обработки результатов опросов общественного мнения. Она состоит в разбивке полной совокупности результатов наблюдений на сравнительно однородные (с малой $внутренней$ дисперсией) слои.

Применительно к интересующей нас задаче проведем расслоение выборки имитируемых времен ожидания по значениям определяющих это ожидание факторов и запишем условное математическое ожидание $\bar{w}$ через найденные в слоях $\{w_k\}$:

$$\bar{w} = \sum_k p_k w_k,$$

где $\{p_k\}$ — вероятности принадлежности результата к k-му слою.

Применяя к слагаемым формулу дисперсии произведения независимых случайных величин, имеем для дисперсии искомой оценки

$$D[\bar{w}] \;=\; \sum_k \{D[p_k]D[w_k] + p_k^2 D[w_k] + w_k^2 D[p_k]\}$$

$$= \sum_k p_k^2 D[w_k] + \sum_k D[p_k](w_k^2 + D[w_k]).$$

Точное знание вероятностей слоев $\{p_k\}$ сводит к нулю взвешенную сумму послойных вторых моментов длительности ожидания и значительно уменьшает $D[\bar{w}]$. Эта формула подсказывает организацию математического эксперимента: определить (предпочтительно аналитически) вероятности $\{p_k\}$ и — отдельно для каждого слоя — получить имитацией первые и вторые моменты времени ожидания.

Успех расслоения обусловливается двумя требованиями:

- стратифицирующая переменная должна быть коррелирована с откликом (результатом);
- должны быть известны вероятности попадания в слои.

Последнее затруднение снимается, если проводить *апостериорную* стратификацию и определять вероятности как $p_k = N_k/N$. При этом число наблюдений в каждом классе должно быть не менее 20, так что малочисленные классы следует объединять.

Наиболее естественной областью применения расслоенных выборок к имитации СМО является учет переменных внешних условий (изменение интенсивностей потоков, длительностей обслуживания, числа действующих каналов и т. п.). В частности, представляется целесообразным моделировать n-канальную СМО с ненадежными каналами как обычную систему с числом каналов от 1 до n и затем взвесить результаты с учетом распределения числа исправных каналов, полученного расчетом или моделированием процесса отказов и восстановлений. Ограничением этого подхода является требование существования стационарного режима в каждом слое.

4.6.2. Определение вероятностей редких событий

Условная имитация является единственным средством получения достаточно точных оценок вероятностей редких событий (например, отказа высоконадежных систем). Каждое такое событие как правило является следствием цепочки порождающих его элементарных. Для решения такой задачи искомую вероятность записывают как сумму произведений условных вероятностей отказа или других неблагоприятных событий

(предполагается, что каждая из этих вероятностей существенно больше искомой). Затем вероятности звеньев цепочек определяются на серии «условных» моделей, а при возможности — аналитическими методами. Наконец, они подставляются в исходную формулу. Практическое решение задач этого класса заметно упрощается предположением, что редкие события случаются *единственным* (наиболее вероятным) образом.

Попытаемся оценить потенциальный выигрыш в числе испытаний на простейшем условном примере. Пусть искомая вероятность

$$P = \prod_{i=1}^{3} p_i, \tag{4.6.1}$$

причем составляющие $\{p_i\}$ имеют порядок 0.01. Тогда сама P имеет порядок 10^{-6}, и для ее *прямого* определения с относительной точностью 10^{-1} согласно (3.5.4) требуется $k/(10^{-6} \cdot (10^{-1})^2) = k \cdot 10^8$ испытаний, где коэффициент k — обсуждавшаяся выше функция доверительной вероятности.

При реализации формулы (4.6.1) относительные погрешности вероятностей $\{p_i\}$ будут складываться, и для расчета каждой из них потребуется $k/(10^{-2} \cdot (10^{-1}/3)^2) = k \cdot 9 \cdot 10^4$ испытаний. Выигрыш по числу испытаний будет в $10^4/(3 \cdot 9) \approx 370$ раз (для сохранения *доверительной вероятности* придется пойти на некоторое его уменьшение).

В целом для успеха условной имитации требуется больше знаний о моделируемом процессе и методологии моделирования, а также умений ими воспользоваться. Вопросы построения с помощью логико-вероятностных методов набора цепочек промежуточных событий, приводящих к катастрофическим финальным, рассматриваются в [34].

Контрольные вопросы

1. В чем состоит общая идея МПД? Почему они сохраняют актуальность, несмотря на рост быстродействия ЭВМ?
2. Расскажите о выборе оценок с наименьшей дисперсией. Какие факторы влияют на эффективность этого метода?
3. Опишите и сопоставьте варианты метода противоположных (дополняющих) выборок.

4. Опишите идею и расчетную схему метода контрольных переменных.
5. Когда применяются параллельные выборки?
6. Опишите идею условной имитации применительно к расслоенным выборкам.
7. Как применить условную имитацию к определению вероятностей редких событий? Опишите методику оценки выигрыша в числе испытаний.

Глава 5

Автоматизация имитационного моделирования

5.1. Потребности и средства

При всей очевидности и простоте основной идеи ИМ разработка программы достаточно сложной модели из-за обилия элементов, многоообразия связей между ними, богатой логики, взаимодействия между проходящими в параллельных ветвях алгоритма процессами и переменными становится нетривиальной и весьма трудоемкой задачей. Поэтому уже в первое десятилетие применения ЭВМ наряду с системами автоматизации программирования численных расчетов (Фортран, Алгол) появились и системы автоматизации имитационного моделирования (САИМ) — в наши дни с возможностями автоматической генерации моделей, интерфейса со сбором данных, обработки выборок большой размерности, распределенной обработки, специализированных языков диалога, ввода-вывода, документирования.

В САИМ *для профессионалов* набольшее внимание уделяется вопросам построения и отладки модели, максимальной гибкости и разнообразию, проверке адекватности, вариабельности техники и технологии ИМ.

В САИМ *для массового использования* на первый план выходят простота и наглядность системы, снижение квалификационных требований к потенциальным пользователям.

К САИМ *для корпоративных клиентов* (заказным) добавляется специфика заказчика: особый интерфейс, привязка к существующим БД, типам объектов, терминологии, графическому оформлению и форматам данных.

Наивысшей фазой развития САИМ являются *интегрированные среды* — в частности, с реализацией *распределенного* моделирования.

5.2. Система AnyLogic

AnyLogic [1, 11] —- одна из немногих российских разработок в области ИМ, получивших признание за рубежом. *AnyLogic* представляет собой среду для графического создания моделей с использованием объектно-ориентированного языка Java. Основным элементом модели в среде *AnyLogic* является *активный объект*. Он имеет внутреннюю структуру и поведение, а также может содержать другие активные объекты как свои элементы.

AnyLogic имеет развитый базовый язык дискретного и смешанного моделирования, на основе которого построены включенные в состав продукта решения для конкретных областей: библиотеки `Enterprise`, `Material Flow`, `Healthcare`. Все они содержит традиционный инструментарий: очереди, задержки, конвейеры, ресурсы и т. п., так что модель и анимация строятся по технологии `drag-and-drop`.

Агенты *AnyLogic* могут динамически создаваться и уничтожаться, перемещаться, общаться друг с другом; иметь знания, стратегию, поведение. При помощи агентов моделируются рынки (агент — потенциальный покупатель), конкуренция, цепочки поставок, ход избирательной компании. Таким образом можно получить представление об общем поведении системы, исходя из предположений о поведении ее элементов — при неизвестных глобальных законах.

Описание поведения объектов производится посредством фрагментов кода на языке Java в специальных полях свойств элементов.

5.3. GPSS World

5.3.1. Общая характеристика

Ни один из языков моделирования не оказал на имитацию столь большого воздействия, как GPSS (General Purpose Simulation System — система имитационного моделирования общего назначения). Созданная Дж. Гордоном в фирме IBM в начале 1960-х гг., она является самым популярным в мире инструментом имитации и наряду с Фортраном и системой TEX являет редчайший пример долгожительства в мире программного обеспечения.

В книге Дж. Шрайбера [41] рассмотрена история создания GPSS. Тон всей книге и характер обсуждаемых приложений задает ее первая глава: «Вопросы моделирования систем массового обслуживания». Создание моделей проиллюстрировано на 27 примерах; предложено свыше 300 задач с решениями либо указаниями. Дополнительно можно рекомендовать учебные пособия [36, 37, 38]. Составление типовых моделей теории очередей для GPSS обсуждается в главе 16 [30] и в [19].

Версии GPSS в России поставляются казанской фирмой «Элина компьютер» — официальным дистрибьютором системы GPSS World. Фирмой создан расширенный текстовый редактор с графическим интерфейсом, позволяющим задать программу в формате иерархии блок-схем, которые автоматически преобразуются в GPSS-программу. Базовый язык GPSS дополнен языком программирования PLUS (Programming Language Under Simulation), близким к традиционным алгоритмическим. На нем можно записывать как выражения для операндов блоков, так и процедуры пользователя, в том числе содержащие циклы и разветвления. Это практически сняло ограничения на формируемые результаты и способы их обработки.

Модель GPSS состоит из сети *блоков*, представляющих необходимые действия или задержки *транзактов*, которые последовательно проходят через блоки. Например, блок `GENERATE` вводит в модель новые транзакты, воспроизводя *рекуррентный* поток заявок с требуемым распределением интервалов между смежными заявками. Транзакты могут представлять собой клиентов, покупателей, ремонтируемые телевизоры, телефонные звонки, электронные сигналы и т. п. В близком *GPSS* языке *DRUGSIM* транзактами являются наркоманы, проходящие разнообразное обслуживание в медицинских и правоохранительных органах.

Транзакты перемещаются в системных времени и пространстве, переходя от одного блока модели к другому и воздействуя на них. Транзакты возникают и уничтожаются, могут расщепляться и сливаться. Входя в блок, транзакт вызывает определяемую типом блока подпрограмму, которая обрабатывает соответствующее событие. Далее транзакт в общем случае пытается войти в следующий блок. Продвижение продолжается до тех пор, пока не окажется, что очередной блок должен выполнить одну из следующих функций:

- удалить транзакт из модели;
- временно заблокировать его в предыдущем блоке до выполнения некоторых условий;
- задержать его на определяемое моделью время.

Тогда начинается продвижение другого транзакта и т. д. — до завершения моделирования. За один шаг работы управляющей программы производится поочередный просмотр всех имеющихся процессов и имитация тех, которые могут быть запущены. При этом может освободиться ранее занятый ресурс или оказаться порожденным процесс более высокого приоритета. Тогда просмотр списка процессов можно начать сначала.

Описание траектории транзакта содержит порядок и имена используемых им «устройств» (одноканальных приборов обслуживания) или «памятей»; временны́е задержки; логические условия, управляющие продвижением транзактов; точки маршрута, в которых производится сбор данных об ожидании или контролируется время прохождения, и т. п. Параметры транзакта на траектории могут менять свои значения, оказывать влияние на маршрут и собираемую статистику.

Каждая из названных функций имеет специальное графическое изображение. Латинские буквы означают последовательные операнды блоков. Многие блоки являются парными (сопряженными). Соответственно их графические представления суть взаимные зеркальные отражения. Из этих блоков составляется схема $GPSS$-модели, описывающая траекторию транзактов (рис. 5.1). Разработчик модели должен так подобрать последовательность блоков и команд, чтобы поведение транзактов соответствовало работе реальной или предлагаемой системы.

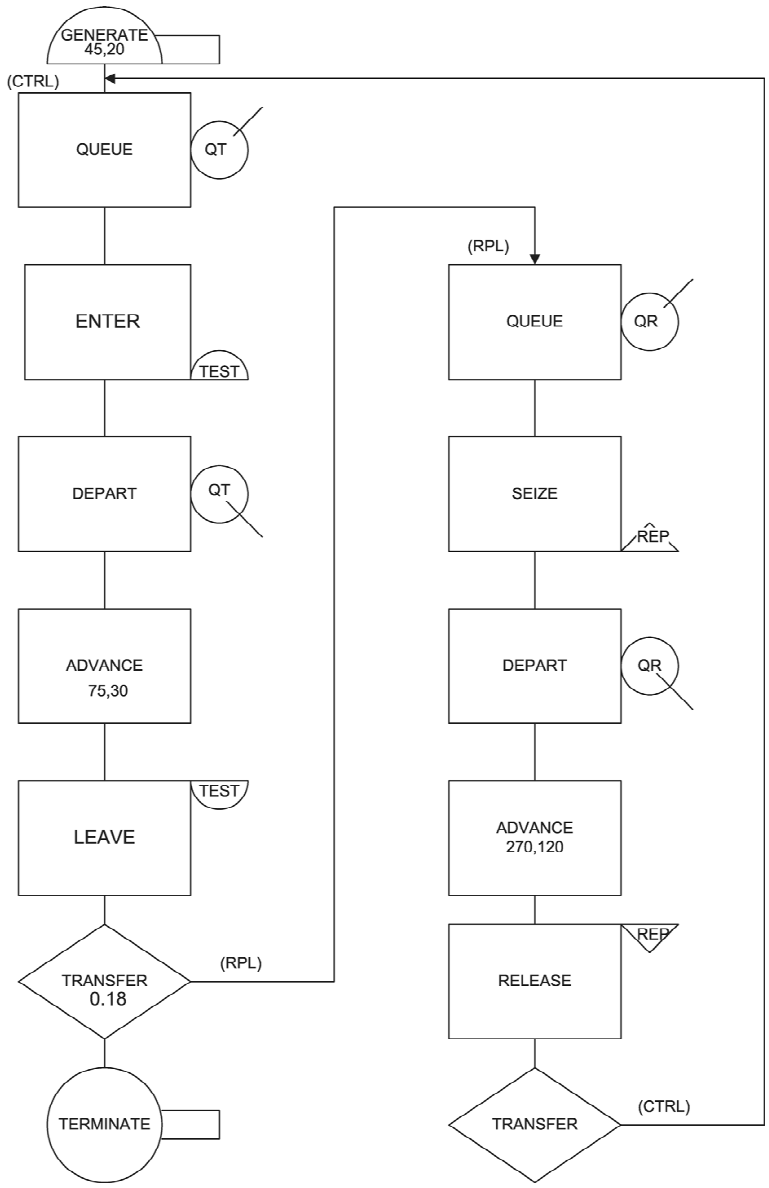

Рис. 5.1. Схема GPSS-модели

Например, при входе транзакта в блок SEIZE он «завладевает» устройством (Facility), имитирующим *одноканальную* систему обслуживания. Интерпретация сущностей, транзактов, параметров и т. п., выбор единицы времени — дело разработчика. Все это надо фиксировать в *словаре* модели.

Траектория и задержки могут зависеть от ситуации в системе и от параметров транзакта. Маршруты следования транзактов, временны́е характеристики, условие прекращения моделирования, требуемые показатели эффективности задаются пользователем.

При описании этих процессов и сборе итоговой статистики используются таймер модельного времени, стандартные числовые атрибуты (СЧА) и параметры транзактов, а также определяемые пользователем переменные, выражения и таблично задаваемые функции. Наиболее сложные аспекты модели — планирование предстоящих событий и очередности их обработки — автоматически реализует планировщик событий (интерпретатор) GPSS.

Динамика модели может отслеживаться как в числовой, так и в графической форме в окнах 20 разных типов — в зависимости от класса отображаемых сущностей, причем в двух режимах: обзорном и детальном. Степень загруженности ресурсов показывается цветом, можно наблюдать за перемещением активного транзакта. В окна можно выводить СЧА и накопленную статистику. Имеются средства интерактивного ввода команд, контрольных остановов и трассировки. Результаты моделирования могут быть автоматически представлены в виде гистограмм соответствующих распределений.

Динамические окна могут быть открыты только после компиляции модели командой

`Window/Simulation Window/<тип_окна>`

Одновременно могут быть открыты несколько окон. Открытие online-окон замедляет моделирование, поскольку системе приходится ждать их обновления. Приведем данные сравнительные данные по длительности прогонов одной и той же простейшей модели:

- все окна закрыты — 4 с;
- выводится таймер — 97 с;
- открыты окно блоков и таймер — 5271 с (почти полтора часа).

Та же модель на *Фортране* была реализована за 0.05 с.

При успехе компиляции активируются интерактивные команды и становятся доступными окна имитации. Далее через меню Command Start задается начальное значение счетчика, которое вместе с имеющим ненулевой декремент блоком TERMINATE определяет длительность моделирования. При запуске модели в статусной строке появляется

Simulation in Progress.

Ниже представлен пример $GPSS$-программы моделирования простейшей одноканальной системы массового обслуживания — с показательно распределенными интервалами распределения интервалов между смежными заявками и длительностей обслуживания.

```
**********************************
*        M/M/1 MODEL             *
**********************************
;       Основной сегмент
1  GENERATE   (Exponential(1,0,1))   ; Генерация заявок
2  QUEUE      Wait                   ; Начало ожидания
3  SEIZE      Chan                   ; Попытка занятия канала
4  DEPART     Wait                   ; При успехе - конец ожидания
5  ADVANCE    (Exponential(2,0,0.9)) ; Задержка на время обслуж.
6  RELEASE    Chan                   ; Освобождение канала
7  TERMINATE                         ; Конец истории транзакта
;     Сегмент таймера
8  GENERATE   50000,,,1              ; Момент окончания 9
   TERMINATE 1                       ; Конец моделирования
```

Здесь комментарии даны по-русски исключительно для первого знакомства с системой: программа, содержащая кириллические символы (в том числе в комментариях и в выводимых строках), работать не будет.

Модель состоит из двух сегментов. В каждом сегменте модели имеется отдельная пара блоков GENERATE — TERMINATE, но только один из TERMINATE может задаваться с операндом, вычитаемым из счетчика завершений. Начальное значение последнего загружается «с пульта» командой START — в данном случае с операндом 1. При обнулении счетчика моделирование прекращается.

Первый сегмент описывает «биографию» направляемой на обслуживание заявки. Во втором генерируется единственный (см. четвертый

операнд GENERATE) транзакт, который обеспечивает завершение моделирования при значении таймера модели 50000 (первый операнд). Отдельный сегмент таймера убыстряет работу модели, так как не требует проверки числа обслуженных заявок для каждого транзакта. Кроме того, этот вариант позволяет при необходимости организовать *заключительную* обработку, программируемую непосредственно перед TERMINATE 1 и выполняемую однократно после завершения сбора статистики.

Теперь обсудим первый сегмент. В нем GENERATE формирует неограниченное число заявок через экспоненциально распределенные моменты времени, доставляемые встроенной функцией EXPONENTIAL. Первый операнд функции указывает номер используемого датчика равномерно распределенных псевдослучайных чисел (1), второй — смещение экспоненты (нулевое), третий — ее среднее значение (единица).

Вошедшая заявка отмечается в очереди QUEUE по имени Wait и пытается захватить (SEIZE) устройство обслуживания по имени Chan. Если устройство занято, транзакт помещается в связанную с ним цепь задержки, которая просматривается в моменты освобождения устройства.

Если устройство свободно, то блок DEPART фиксирует конец пребывания в очереди Wait. Соответственно накапливается статистика ожидания, связанная с этой очередью. Отметим, что блоки QUEUE и DEPART определяют только *необходимость сбора статистики* по данной очереди: их отсутствие не влияет на реальное существование и динамику очередей.

Захватившая устройство заявка «отбывает» в нем (блок ADVANCE) интервал с показательно распределенной длительностью и средней задержкой 0.9. Для его формирования используется другой (второй) ДСЧ. Таким образом, коэффициент загрузки системы $\rho = \lambda b/n = 1.0 * 0.9/1 = 0.9$. По истечении задержки блок RELEASE освобождает канал Chan, и траектория транзакта заканчивается *без воздействия на счетчик завершений*.

Обращают на себя внимание простота и логичность обсуждаемого текста в сравнении с программой модели на алгоритмическом языке общематематического назначения из раздела 1.6 — по крайней мере в этом элементарном случае. Отметим, однако, необходимость знания многих технических деталей (в первую очередь связанных с позицией и смыслом аргументов блоков), так что упомянутая простота обходится недешево.

5.3.2. Тестирование GPSS World

Общим правилом работы с новым программным обеспечением математического характера является его тестирование на задачах с известным решением (предпочтительно аналитическим). Для системы $M/M/1$ теоретическая средняя длина очереди $q = \rho^2/(1-\rho) = 0.9^2/(1-0.9) = 8.1$. Среднее время ожидания w в соответствии с формулой Литтла (закон сохранения стационарной очереди) должно быть равно q/λ и в данном случае ($\lambda = 1$) численно совпадать со средней длиной очереди. Теоретическая вероятность незанятости системы равна $1 - \rho = 0.1$. Следовательно, в среднем десятая часть заявок должна получать обслуживание без ожидания. Сопоставим с этим предположением результаты моделирования из таблицы 5.1.

Таблица 5.1. Тестирование GPSS/W на модели $M/M/1$

Показатель	Теория	Число испытаний		
		50 000	200 000	500 000
Коэффициент загрузки	0.900	0.896	0.897	0.899
Средн. время обслуживания	0.900	0.899	0.902	0.901
Число входов		49843	199042	499032
Из них с нулевым ожиданием		5340	21001	50691
Средняя длина очереди	8.100	5.132	5.757	6.690
Среднее время ожидания	8.100	5.148	5.785	6.703

Формула Литтла и доля заявок, принимаемых к обслуживанию без ожидания, подтверждаются с достаточно высокой точностью. Это дает основания доверять интерпретатору GPSS/W. Показатели, определяемые только *средними* значениями моделируемых первичных величин (среднее время обслуживания и коэффициент загрузки), также вполне приемлемы.

Результаты, связанные с ожиданием, оставляют желать лучшего. Причина может быть только одна: недостаточно качественный генератор псевдослучайных чисел. Увеличение числа проведенных испытаний в общем приближает результаты к ожидаемым, но даже при 500 тыс. испытаний погрешность составляет около 17%.

5.3.3. Оценка GPSS World

GPSS/W является весьма ценным инструментом имитационного моделирования: свободным от ограничений аналитических и численных методов, достаточно «прозрачным», допускающим нестандартную обработку данных и снимающим с программиста множество нетривиальных проблем программирования и отладки моделей. Тем не менее приходится отметить наличие у нее ряда серьезных недостатков:

1. Громоздкость системы и явную перегруженность встроенными возможностями (многообразие примитивов).

2. Медленная работа интерпретатора.

3. Отсутствие концептуального единства. Достаточно указать, например, различие в обращении к элементам матрицы при простой ссылке и изменении значения, круглые индексные скобки в основном тексте и квадратные — в PLUS-выражениях; обязательность приставки `MX` при ссылке на глобальную матрицу в тексте модели и столь же обязательное ее отсутствие внутри процедур; контекстно зависимый вид ссылок на параметры активного транзакта.

4. Неудачные обозначения операторов отношения `L, G, E` (было бы лучше согласовать с фортрановскими); арифметическое `SQR` используется для квадратного корня (в Паскале так обозначается *квадрат*); состояние логических ключей описывается как `SET` и `RESET` (буквальный перевод «установлен» и «установлен заново») вместо `ON`, `OFF`.

5. Однократность прерываний «устройств» и недопустимость прерываний для «памятей», которые могут быть использованы для моделирования многоканальных устройств обслуживания и, следовательно, окажутся подвержены прерываниям.

6. Отсутствие средств подбора параметров теоретических вероятностных распределений по заданным моментам.

7. Невозможность менять тип шкал графиков (только линейные) и их разметку, цвет и структуру линий, что может сделать их неотличимыми друг от друга и/или от фона при черно-белом выводе. Аргументом графиков может быть только время, так что распределение вероятностей состояний системы автоматически построить нельзя. Кстати, для такого графика обязательна *логарифмическая* шкала.

Этот перечень можно было бы продолжить.

Заключение

Имитационное моделирование позволяет оценивать работу практически любых систем и сетей обслуживания, что делает его (как поставщика первичных данных и эталонных результатов) незаменимым помощником при разработке новых теоретических методик и универсальным инструментом решения практических задач.

Имитационное моделирование все шире внедряется в проектирование производственных и иных организационных процессов и оперативного управления ими, в разработку тренажеров. Наличие имитационной модели в комплекте документов проектирования или модернизации нового производства либо технологического процесса становится обязательным. В США на ИМ тратятся десятки млрд долларов в год. Имитационное моделирование принимаемых решений, проектов развития и технологий постоянно применяется такими компаниями, как Boeing, Compaq, Xerox, IBM, Intel, Lockheed, Motorola, General Motors, Ford, Standard Oil, Cray Research и многими другими, а также рядом правительственных организаций (Агентство национальной безопасности, ВВС, ВМФ, NASA).

Острейшая необходимость внедрения математического моделирования во все сферы экономики, управления, военного дела, общественной жизни осознаны, наконец, и в России. Имитационным моделированием серьезно интересуются в металлургии, нефтегазовой отрасли, производстве стройматериалов и пищевых продуктов; в государственном, военном и корпоративном управлении.

Достоинствами ИМ являются:

- возможность моделирования любого марковского процесса обслуживания (термин *марковский* означает, что будущее поведение процесса полностью определяется его текущим состоянием — условие, которое обычно подразумевается, но редко оговаривается явно);

- легкость учета особенностей ситуаций, практически не поддающихся численному анализу: многоканальные приоритетные системы; системы с немарковским нетерпением; нестационарные задачи — через получение временны́х сечений процесса по множеству реализаций; процессы типа `FORK AND JOIN` и др.;
- наглядность реализующей модель программы (при необходимости — и процесса реализации), легкость интерпретации промежуточных результатов, что облегчает отладку;
- простота освоения основных идей и (в большинстве случаев) соответствующих технологий.

Дисциплина «Моделирование систем» введена в учебные планы не только компьютерных, но и ряда экономических специальностей и к тому же превратилась в общепрофессиональную (стала «ближе к массам»). В Интернете появились российские сайты, посвященные имитационному моделированию. В 2011 г. создано Национальное общество ИМ (президент — член-корреспондент РАН Р.М. Юсупов). В процессе своего становления общество провело восемь весьма представительных Всероссийских конференций по имитационному моделированию [21]. К примеру, уже на второй авторами и соавторами докладов были 194 человека: специалисты, предприниматели, менеджеры и научные работники 37 городов России, Украины, Беларуси, Латвии и Германии. Среди них насчитывалось 40 докторов и 41 кандидат наук (технических, физико-математических, военных, медицинских, биологических, экономических, естественных и др.), а также специалисты и 7 студентов: из ЛЭТИ, Политехнического и Кораблестроительного университетов Санкт-Петербурга; МАИ; Томского, Орловского, Белорусского университетов (на ИММОД-2013 адъюнкт ВКА им. Можайского А.В. Уланов был удостоен почетного диплома). На конференциях распространялись написанные Ю.Г. Карповым, Н.Б. Кобелевым, Ю.И. Рыжиковым и В.Н. Томашевским учебники по моделированию, а также демонстрационные версии программных продуктов (AnyLogic, GPSS World). Хотелось бы, однако, подчеркнуть, что исследователи и практики явно недооценивают важность твердого усвоения «имитаторами» *теоретических основ* ИМ: теории вероятностей, математической статистики и теории очередей.

Перечислим особенности *имитационного моделирования*, которые «имитаторы» должны постоянно иметь в виду:

1. Основные недостатки ИМ:

- каждый прогон ИМ — это частный результат, по ним труднее делать обобщения;
- ввиду неидеальности используемых датчиков псевдослучайных равномерно распределенных чисел надежды на принципиальную возможность добиться сколь угодно высокой точности увеличением числа испытаний безосновательны;
- получаемые оценки искомых величин имеют статистическую погрешность;
- имитационные модели неудобны для оптимизации, поскольку выигрыш в значении целевой функции может быть перекрыт статистическими и инструментальными погрешностями;
- прогон модели как правило имеет бóльшую трудоемкость, чем ее численный обсчет (если такая возможность имеется);
- относительная погрешность определения вероятностей редких событий велика и зачастую не может быть снижена до приемлемого значения при разумном числе испытаний;
- ветви алгоритма моделирования находятся в сильной взаимной зависимости, что затрудняет обнаружение ошибок и затягивает процесс отладки.

2. Для определения тонких эффектов и вероятностей редких событий ИМ *без дополнительных ухищрений* — не самый подходящий инструмент.

3. Никакая система, специализированная на ИМ, не может быть универсальной уже из-за необходимости встроенного интерпретатора, разработчики которого в принципе не в состоянии предусмотреть *всего*. К сожалению, логика интерпретатора как правило оказывается скрытой, а вмешательство в нее (разумеется, под ответственность исследователя) — невозможным.

4. Платой за претензию на «универсальность» всегда является избыточность модели, которая многократно замедляет ее работу (по опыту применения GPSS World — в десятки раз даже при отключенном выводе динамических параметров).

5. Важнейшим достоинством автоматизированных систем ИМ является ускорение разработки и в особенности отладки программ моделирования. Однако, по глубокому убеждению автора, это не дает carte blanche работающим «без понятия» так называемым «непрограммирующим пользователям». *Реальной замены знанию не существует*, и его отсутствие рано или поздно скажется.

6. Встроенные средства *оптимизации имитационных моделей* непомерно трудоемки. Более того, из-за дискретности числа обслуживающих устройств и рядов их промышленных образцов, ограниченных возможностей управления траекториями заявок и дисциплинами обслуживания и зашумленности результатов статистическими погрешностями они практически бесполезны. Оптимизация модели должна проводиться в *диалоговом* режиме работы с ней путем последовательной расшивки «узких мест» — уменьшением потоков через наиболее загруженные узлы, увеличением числа каналов или их быстродействия.

7. Идеи ИМ настолько просты, что работы в этой области (особенно с применением моделирующих сред) обладают «малой наукоемкостью»: очень трудно предложить технологическую новинку, *не очевидную для специалиста* (стандартное требование диссертационных советов и журнальных редколлегий). Поэтому ИМ следует рассматривать главным образом как инструмент для решения практических задач большой размерности и как последнее средство — для исследования особо сложных задач (в первую очередь нестационарных, многоресурсных, многоагентных, с расщеплением и слиянием заявок); проверки приемлемости аналитических допущений; максимального приближения к реальности. Крайне желательно применение методов понижения дисперсии.

8. Обучение имитационному моделированию должно начинаться с составления или по крайней мере с разбора моделей, записанных на языках программирования общематематического назначения, т. е. с *«прямой» имитации*. Нельзя применять САИМ, не имея никакого понятия о логике работы типовой модели. Использование прямой имитации предпочтительно и в *исследовательских* работах.

Отдельного обсуждения обслуживает сопоставление «авторского» моделирования (например, на Фортране) и работы в моделирующих средах (GPSS World, AnyLogic и т. п.). Такие среды в современном исполнении предоставляют пользователю удобный интерфейс вплоть до графической сборки программы, а также средства визуализации процесса и его результатов. Это ускоряет подготовку и проведение имитационных

экспериментов. Следует, однако, иметь в виду, что все такие системы имеют встроенный и защищенный от изменений интерпретатор программы, который *принципиально не является универсальным* и для которого нельзя дать исчерпывающий перечень того, чего он *не может делать*. Кроме того, стремление разработчиков к расширению применимости приводит к непомерной избыточности порождаемых программ и как следствие — к замедлению их работы.

Хочется надеяться, что изучение этой книги поможет читателю понять возможности и технологии имитационного моделирования — и заняться умной и полезной работой (вместо *имитации* таковой).

Литература

[1] Аксенов, К.А. Динамическое моделирование мультиагентных процессов преобразования ресурсов /К.А. Аксенов, Н.В. Гончарова. — Екатеринбург: УГТУ-УПИ, 2006. — 311 с.

[2] Бусленко, Н.П. Метод статистических испытаний и его реализация на электронных цифровых машинах /Бусленко Н.П., Шрейдер Ю.А. — М.: Наука, 1962.

[3] Бусленко, Н.П. Моделирование сложных систем /Бусленко Н.П. — М.: Наука, Физматгиз, 1979. — 400 с.

[4] Вадзинский, Р.Н. Справочник по вероятностным распределениям /Р.Н. Вадзинский — СПб.: Наука, 2001. — 295 с.

[5] Городецкий, В.И. Математическое программирование и массовое обслуживание: учеб. пособие /В.И. Городецкий, Ю.И. Рыжиков. — Л.: ВИКИ им. А.Ф. Можайского, 1975. — 182 с.

[6] Городецкий, В.И. Мультиагентные технологии для оперативного управления ресурсами в реальном масштабе времени /Городецкий В.И. и др. //Доклад на пленарном заседании мультиконференции по информационным технологиям в управлении 12-14 октября 2010 г. — СПб.: ЦНИИ «Электроприбор».

[7] Девятков, В.В. Методология и технология имитационного исследования сложных систем: учебник /В.В. Девятков. — М.: ИНФРА-М, 2013. — 448 с.

[8] Девятков, В.В. Имитационные исследования в среде моделирования GPSS STUDIO: Учебн. пособие /В.В. Девятков, Т.В. Девятков, М.В. Федотов. — М.: Вузовский учебник: ИНФРА-М, 2018. — 283 с.

[9] Задорожный, В.Н. Аналитико-имитационные исследования систем и сетей массового обслуживания: монография /В.Н. Задорожный. — Омск: Изд-во ОмГТУ, 2010. — 280 с.

[10] Ивашкин, Ю.А. Агентные технологии и мультиагентное моделирование: Учебн. пособие /Ю.А. Ивашкин — М.: МФТИ, 2013. — 267 с.

[11] Карпов, Ю.Г. Имитационное моделирование систем. Введение в моделирование с AnyLogic 5 /Ю.Г. Карпов. — СПб.: БХВ-Петербург, 2005. — 400 с.

[12] Кендалл, М.Дж. Теория распределений /М.Дж. Кендалл , А. Стьюарт. Пер. с англ. — М.: Наука, 1966. — 587 с.

[13] Кельтон, В. Имитационное моделирование /В. Кельтон, А. Лоу. — Пер. с англ., 3-е изд. — СПб.: Питер, Киев, БХВ, 2004. — 847 с.

[14] Клейнен, Дж. Статистические методы в имитационном моделировании /Дж. Клейнен. — Пер. с англ. — М.: Статистика, 1978. — Вып.1, 221 с.; вып.2, 386 с.

[15] Кнут, Д. Искусство программирования для ЭВМ. Т. 2. Получисленные алгоритмы /Кнут Д. — Пер. с англ. — М.: Мир, 1977. — 724 с.

[16] Кобелев Н.Б. Имитационное моделирование: Учебное пособие /Кобелев Н.Б., Половников В.А., Девятков В.В. — М.: НИЦ ИНФРА-М, 2013. — 368 с.

[17] Корн, Г. Справочник по математике для научных работников и инженеров /Г.Корн, Т. Корн. — Пер. с англ. — М.: Наука, 1973. — 831 с.

[18] Крейн, М. Введение в регенеративный анализ моделей /М. Крейн, О. Лемуан. — Пер. с англ. — М.: Наука, 1982. — 104 с.

[19] Кудрявцев, Е.М. GPSS World: основы имитационного моделирования различных систем /Е.М. Кудрявцев. — М.: ДМК Пресс, 2004. — 320 с.

[20] Мандельброт, Б. Фрактальная геометрия природы /Б. Мандельброт. — Пер. с англ. — М.: Институт компьютерных исследований, 2002. — 656 с.

[21] Материалы Всероссийских научно-практических конференций «Опыт практического применения языков и программных систем имитационного моделирования в промышленности и прикладных разработках». — СПБ: ЦНИИ технологии судостроения, 2003–2017 гг.

[22] Ослин, Б.Г. Имитационное моделирование систем массового обслуживания: учеб. пособие /Б.Г. Ослин. — Томск: изд-во Томского политехнического ун-та, 2010. — 128 с.

[23] Петров, А.А. Экономика модели. Вычислительный эксперимент /А.А. Петров. — М.: Наука, 1996. — 250 с.

[24] Плотников, А.М. Анализ современного состояния и тенденции развития имитационного моделирования в Российской Федерации /А.М. Плотников, Ю.И. Рыжиков, Б.В. Соколов, Р.М. Юсупов //Труды СПИИРАН, вып. 2(25). — СПб.: 2013, с. 42–112.

[25] Русаков, А.М. Исследование и моделирование сложных систем: Учебн. пособие /А.М. Русаков. — М.: Гос. ун-т приборостроения и информатики, 2014. — 90 с.

[26] Рыжиков, Ю.И. Эффективность и эксплуатация программного обеспечения ЭЦВМ: Учебн. пособие /Ю.И. Рыжиков. — МО СССР, 1985. — 263 с.

[27] Рыжиков, Ю.И. Имитационное моделирование систем массового обслуживания: Учебн. пособие /Ю.И. Рыжиков. — Л.: ВИККИ им. А. Ф. Можайского, 1991. — 111 с.

[28] Рыжиков, Ю.И. Решение научно-технических задач на персональном компьютере /Ю.И. Рыжиков. — СПб.: КОРОНА принт, 2000. — 271 с.

[29] Рыжиков, Ю.И. Теория очередей и управление запасами /Ю.И. Рыжиков. — СПб.: Питер, 2001. — 376 с.

[30] Рыжиков, Ю.И. Имитационное моделирование. Теория и технологии /Ю.И. Рыжиков. — СПб.: КОРОНА принт, 2004. — 380 с.

[31] Рыжиков Ю.И. Имитационное моделирование: Курс лекций /Ю.И. Рыжиков. — СПб.: ВКА им. А. Ф. Можайского, 2007. — 125 с.

[32] Рыжиков, Ю.И. Компьютерное моделирование систем с очередями: Курс лекций /Ю.И. Рыжиков. — СПб.: ВКА им. А. Ф. Можайского, 2007. — 164 с.

[33] Рыжиков, Ю.И. Современный Фортран: Учебник /Ю.И. Рыжиков. — СПб.: КОРОНА принт, 2004. — 288 с.

[34] Рябинин, И.А. Надежность и безопасность структурно-сложных систем /И.А. Рябинин. — СПб.: Политехника, 2000. — 248 с.

[35] Салеев, В.М. Библиотека алгоритмов и программ генерации и статистической обработки последовательностей случайных чисел: материалы по программному обеспечению /В.М. Салеев. — Минск: Ин-т математики АН БССР, 1986. — 138 с.

[36] Советов, Б.Я. Моделирование систем: Учебник для вузов /Б.Я. Советов, С.А. Яковлев. — М.: Высшая школа, 2001. — 343 с.

[37] Советов, Б.Я. Моделирование систем: Практикум /Б.Я. Советов, С.А. Яковлев. — М.: Высшая школа, 1999. — 224 с.

[38] Томашевский, В.Н. Имитационное моделирование в среде GPSS /В.Н. Томашевский, Е.Г. Жданова. — М.: Бестселлер, 2003. — 416 с.

[39] Шеннон, Р. Имитационное моделирование систем — искусство и наука /Р. Шеннон. — Пер. с англ. — М.: Мир, 1978. — 418 с.

[40] Шнепс, М.А. Численные методы теории телетрафика /М.А. Шнепс. — М.: Связь, 1974. — 232 с.

[41] Шрайбер, Т.Дж. Моделирование на GPSS /Т.Дж. Шрайбер. — Пер. с англ. — М.: Машиностроение, 1980. — 592 с.

[42] Bratley P., Fox B.L., Schrage L.E. A Guide to Simulation. — N.Y.: Springer Verlag, 1983. — 383 pp.

[43] Chan J.C.C, Kroese D.P. Rare-Event Probability Estimation wit Conditional Monte Carlo //Ann. Oper. Res., v. 189, 2011. — P. 43–61.

[44] Devroye L. Non-Uniform Random Variate Generation. — N.Y.: Springer Verlag, 1986. — 843 pp.

[45] Iglehart D.L., Shedler G.S. Regenerative Simulation of Response Times in Networks of Queues. — Berlin: Springer Verlag, 1980. — 204 pp.

[46] Xia Li, Cao Xi-Ren. Performance Optimization of Queueing Systems with Perturbation Realization //European J. of Operat. Res., v.18, 2012. — P. 293-304.

[47] Nance R.E., Sargent R.G. Perspectives on the evolution of simulation // Operations Research. — 2002. — v. 50, No. 1. — P. 161–172.