навыков программной генерации случайных объектов и процессов с заданными вероятностными свойствами и оценки их характеристик.

Выработка указанных навыков является комплексной, трудоемкой задачей. Требует наличия инструментов, позволяющих реализовать типовые этапы имитации процессов в ручном и автоматических режимах, отслеживать ход и результаты имитации, тестировать полученные результаты и т. п. Все указанное может быть реализовано в виде единой системы – фреймворка как набор связанных средств с возможностью многовариантного использования.

Работа является развитием комплекса средств [1]. Система обеспечивает изучение: алгоритмов программной генерации квазиравномерных чисел с использованием конгруэнтных методов (универсальный и частные квадратичный, аддитивный, композитные методы), генераторов Таусворта, Фиббоначи, Мерсенна и др.; оценку качества выборок с помощью частотных, сериальных и др. тестов, проверку гипотез о характере распределения и т. д.; изучение техник имитации случайных объектов с заданными распределениями, включая произвольные, задаваемые табличными аналогами законов распределения, интервальными рядами, выборками и т. д.; изучение алгоритмов генерации процессов, анализа их стационарности и т. д.

В основу изучения указанных процессов и построения системы положен модульный подход. Соответственно рассмотрена возможность обеспечения многовариантного применения комплекса средств (готовых функций, классов, модулей) - от применения его как набора автономных инструментов для “ручного” использования до автоматической поддержки сценариев обучения и тестирования, реализуемых в режиме “конструктора” путем коммутации соответствующей технологической цепочки из элементов системы.

Используемый теоретический аппарат: методы имитационного моделирования дискретных систем [2]; объектно-ориентированный подход, каркасное программирование, средства UML для разработки и реализации системы. Макетирование системы проведено средствами Microsoft Visual Studio.

Список цитированных источников

1. Слинко, Е.В. Электронный лабораторно-практический комплекс / Е.В. Слинко, А.О. Скарубо // Новые математические методы и компьютерные технологии в проектировании, производстве и научных исследованиях: материалы 21 РНК студентов и аспирантов, Гомель, ГГУ им. Ф.Скорины, 19-21 марта – Гомель, 2018. - С. 298-299.

2. Кельтон, В. Имитационное моделирование. Классика CS / В. Кельтон, А. Лоу. – СПб.: Питер, 2004. – 630 с.

УДК 519.863 + 004.588

РАЗРАБОТКА ПРИЛОЖЕНИЯ ДЛЯ РЕШЕНИЯ ЗАДАЧИ О НАЗНАЧЕНИЯХ ВЕНГЕРСКИМ МЕТОДОМ

Ханцевич А. Э., Хомицкая Т. Г., Лизун Л. В.

Брестский государственный технический университет, г. Брест, Беларусь

Введение. В реальной жизни мы часто сталкиваемся с ситуациями, когда необходимо сопоставить объекты одного типа объектам другого типа. Например, команде необходимо реализовать какой-либо проект. Проект разбивается на более

мелкие составляющие, каждая из которых требует своей реализации, отличной от других. Члены команды с разной степенью эффективности могут реализовать необходимые части проекта. Необходимо распределить каждую составляющую между всеми членами команды таким образом, чтобы выполнить проект максимально качественно. Данный тип задачи относится к задачам о назначениях [1, 2]. С целью оптимизации данного типа задач было разработано приложение, демонстрирующее работу алгоритма, предоставляя информацию об этапах и порядке выполнения решения, позволяя быстро найти решение задачи и проверить качество усвоения пользователем данного метода.

Постановка задачи о назначениях. Одной из фундаментальных задач *оптимизации* является *задача о назначениях.* Данный тип задач сводится к следующей формулировке:

Имеется некоторое число *работ* (A_1, A_2, …, A_n) и некоторое число *исполнителей* (B_1, B_2, …, B_n). Любой исполнитель может быть назначен на выполнение любой (но только одной) работы, но с *неодинаковыми* затратами. Эффективность i-го исполнителя на j-й работе определяется мерой эффективности c_{ij}, где i, j = 1, 2, …, n. Нужно распределить работы так, чтобы выполнить работы с *минимальными* затратами.

Примечание: алгоритм решения задачи о назначениях предполагает *минимизацию* ее целевой функции. Если имеется задача о назначениях, целевую функцию которой нужно *максимизировать*, то перед началом решения вводят дополнительный шаг: все элементы исходной матрицы умножают на (*-1*). Дальнейшее решение идентично решению задачи на минимум.

Алгоритм решения и демонстрация работы программы. Алгоритм состоит из предварительного этапа и не более чем (*n*-2) последовательно проводимых итераций, где *n* – размер матрицы.

Предварительный этап включает в себя поиск минимального элемента в каждом из столбцов матрицы с последующим его вычитанием. Затем аналогичное преобразование производится со строками. В результате образуется матрица, эквивалентная исходной, с неотрицательными элементами, в каждой строке и столбце которой имеется по крайней мере один нуль.

Заканчивается предварительный этап построением последовательности *независимых нулей* таким образом, что каждая строка и столбец матрицы содержат не более одного нуля, помеченного как независимый (рисунок 1).

+	+		+		
3	0	8	0*	0	
0*	2	0	2	0	
2	0*	3	3	6	
3	0	6	4	7	
1	0	1	3	4	

Рисунок 1 – Построение последовательности независимых нулей

Дальнейшее решение задачи сводится к проведению ряда последовательных операций, результатом которых будет увеличение числа независимых нулей на единицу (например, построение цепочки 0* и 0') (рисунок 2).

Рисунок 2 – Построение цепочки 0* и 0'

Задача считается решенной, когда каждая строка и столбец будут содержать 0* (рисунок 3).

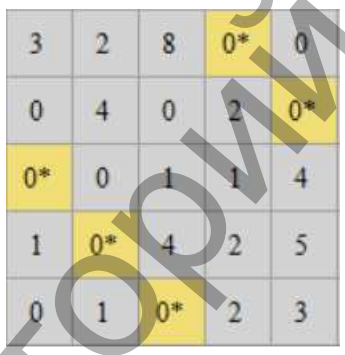

Рисунок 3 – Результат работы программы: установка соответствия между каждой работой и её исполнителем

Затем необходимо установить соответствие между полученной матрицей и исходной. Просуммировав найденные значения, можно получить значение целевой функции и найти минимальные затраты, необходимые на выполнение всех работ (рисунок 4).

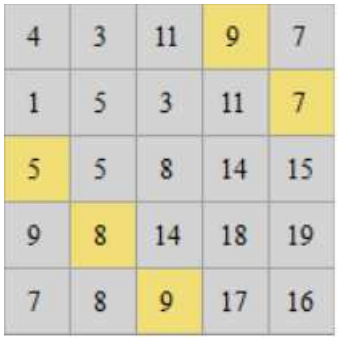

Рисунок 4 – Установка соответствия между полученной матрицей и исходной

В данном случае её значение равняется 38:

$$L(x) = 9 + 7 + 5 + 8 + 9 = 38.$$

Процесс реализации приложения. Приложение разрабатывалось на объектно-ориентированном языке программирования C#. За основу был взят шаблон разработки десктопных приложений Windows Forms, предоставляющий гибкий функционал для создания графического интерфейса приложения, а также паттерн проектирования MVC (Model-View-Controller), предполагающий разделение приложения на отдельные компоненты, такие как:

- Модель – предоставляет собой объект с некоторым набором полей для хранения текущего состояния решения задачи;
- Представление – графическая составляющая приложения. Изменяет состояние модели через контроллер;
- Контроллер – промежуточное звено, позволяющее связать модель с представлением, скрывая детали реализации алгоритма.

Выбранная структура организации приложения позволяет легко изменить (либо же полностью заменить на другие) представление/контроллер, так как модель напрямую с ними не связана.

Приложение имеет 3 режима работы. Обычный режим работы предназначен для быстрого решения искомой задачи. Режим обучения предназначен для пошагового решения задачи с объяснением пользователю каждой итерации алгоритма. Режим «контроль» предназначен для проверки, насколько хорошо пользователь усвоил данный алгоритм решения, и предполагает пошаговое решение задачи за тем исключением, что пользователю необходимо проделывать каждый шаг, а программа будет следить за правильностью его действий. Периодически появляются модальные окна с вопросами по алгоритму, на которые пользователю также необходимо ответить. По завершении данного режима предоставляется статистика, показывающая время, за которое была решена задача, а также ошибки, которые допустил пользователь.

Кроме основных функций, программа предоставляет возможность сохранить исходные данные в виде файла в выбранной директории или загрузить предварительно сохранённый файл с данными.

Практическое применение. Приложение можно использовать для обучения студентов по предметам «Системный анализ и исследование операций», «Дискретная математика и математическое моделирование».

Список цитированных источников

1. Иванов, Б.Н. Дискретная математика. Алгоритмы и программы / Б.Н. Иванов. – М.: Лаборатория Базовых Знаний, 2001. – 288 с.

2. Таха, Х.А. Введение в исследование операций / Х.А. Таха – М.: ИД "Вильямс", 2005. – 912 с.