УДК 004.55

СИСТЕМА ВИЗУАЛЬНО-ИНТЕРАКТИВНОГО ТЕСТИРОВАНИЯ АЛГОРИТМОВ

Садовый Д.В.
к.т.н., профессор Рувинская В.М.
Одесский Национальный Политехнический Университет, УКРАИНА

АННОТАЦИЯ. В работе предложен подход к разработке систем, позволяющих создавать программы, основанные на сложных алгоритмах, и тестировать их, используя визуальные инструменты.

Введение. Для сложных информационных систем, где необходимы высокая производительность, вычислительная мощность и надежность, по сей день слабым местом остаются высокая сложность требуемых алгоритмов и затраты на их разработку. Не менее сложная работа возлагается и на процесс тестирования – убедиться в том, что разработанный алгоритм работоспособный и удовлетворяет поставленным требованиям. Существует множество автоматических систем, такие как *Hewlett Packard software testing framework* [1], *IBM Rational Functional Tester*, которые умеют выполнять модульное, функциональное, регрессионное и другие виды тестирования. Однако не существует специализированных средств для тестирования сложных алгоритмов, в частности, с реализацией математических методов, позволяющих визуализировать объекты и интерактивно с ними работать.

Цель работы. Целью работы является снижение затрат на тестирование алгоритмической составляющей программного обеспечения путем использования визуально-интерактивных средств тестового покрытия решений.

Основная часть работы. Предлагается подход к разработке программных систем, который даст пользователю возможность реализовать требуемый алгоритм на одном из современных языков программирования и в полной мере покрыть его тестовыми данными с помощью интерактивной визуальной сцены.

За основу взят формат международной студенческой олимпиады *ACM-ICPC* [2], в которой соревнуются команды из 3 человек, решая различные математические и алгоритмические задачи. В основном задачи на таких соревнованиях базируются на сложных алгоритмах, описание задачи содержит условие, а также текстовые входные и выходные данные. Проверка корректности решений выполняет автоматическая система тестирования.

В промышленной среде разработки, в отличии от олимпиадной, не все задачи имеют четкого решения, которое можно проверить с помощью автоматических систем. Более того, для олимпиадных задач для проверки решений авторы создают набор тестовых данных. Это означает, что даже для четко определенного решения необходимы затраты на проведение проверки корректности. Чтобы упростить данную часть разработки и сформировалась идея о создании системы для функционального тестирования алгоритмов на основе *Test Case-ов*, которые будут создаваться с помощью визуально-интерактивной графической сцены.

В качестве примера рассмотрим олимпиадную геометрическую задачу по нахождению минимальной выпуклой оболочки по набору точек на плоскости, которая представляет собой минимальное выпуклое множество, их содержащее. Один из возможных алгоритмов решения данной проблемы – алгоритм Грэхема [3]. Задача состоит в следующем: заданы N точек, то есть 2*N чисел во входных данных. На выходе нужно получить точки на выпуклой оболочке. Чтобы проверить правильность работы алгоритма, можно составить *Test Case* вручную, затем на листке бумаги изобразить полученные данные.

На помощь разработчикам приходит разработанная нами система визуального тестирования алгоритмов. Детально процесс тестирования изображен на рис.1. С помощью встроенной графической системы разработчик расставляет объекты (в нашем случае – точки) на графической сцене, тем самым создавая очередной *Test Case*. Затем система автоматически запускает на выполнение разработанный алгоритм, подставляя туда созданные данные. После

завершения работы алгоритма ответ визуализируется на графической сцене. Разработчик имеет возможность визуально оценить качество разработанного алгоритма. В любой момент имеется возможность изменять положения объектов, тем самым генерируя новый набор входных данных. Менее чем за минуту можно сгенерировать более тысячи наборов входных данных, тем самым за короткий промежуток времени покрыть решение большой тестовой выборкой. Протестировать таким способом можно любой алгоритм, входные данные которого можно представить с помощью объектов. Например, мы можем тестировать алгоритмы на графах путем расстановки вершин и связей между ними.

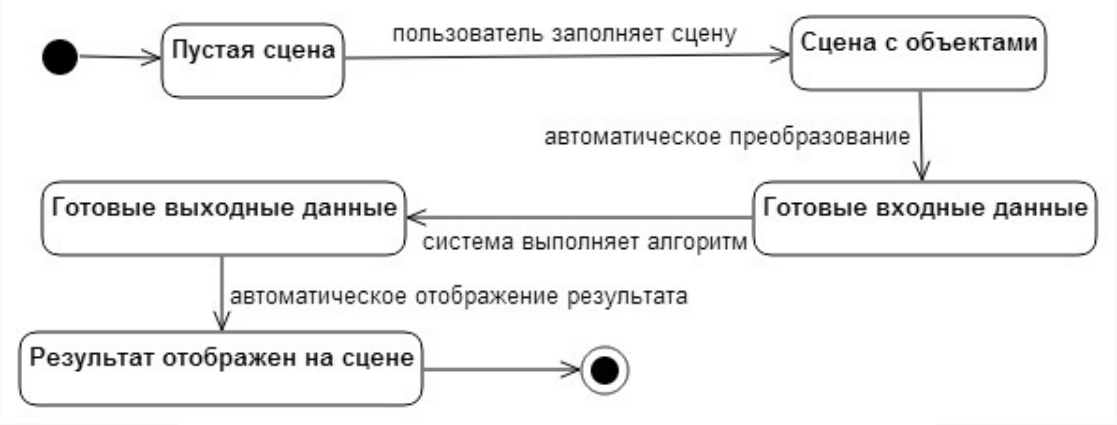

Рисунок 1. Диаграмма состояний процесса выполнения алгоритма

Под преобразованием объектов понимается получение текстового описания входных данных к тестированию. Например, из точек на графической сцене получаем список пар чисел (x, y), обозначающие их положение на декартовой плоскости.

Для обработки *Test Case-ов* в системе предусмотрен набор правил, по которым необходимо написать программу общения с системой. Под общением понимаются запрос данных и отправка команд на отображение. Теперь для проверки алгоритма от программиста нужно лишь разработать алгоритм и написать небольшую программу для общения с системой.

На этапе отображения ответа система получает набор команд к отрисовке. Например, в задаче про выпуклую оболочку командами будут рисование линий между точками полученной фигуры. В результате пользователь может наглядно видеть качество работы алгоритма.

Выводы. В представленной работе описана система для проведения визуально-интерактивного тестирования алгоритмических систем. Недостатком системы является узкая специализация, базирующаяся лишь на тестировании алгоритмических модулей систем. Однако ни одна из существующих систем тестирования не могут дать пользователю возможность максимально полно и наглядно протестировать сложные алгоритмы. С использованием визуально-интерактивной среды вероятность расхождения результатов алгоритма и требований к нему уменьшается в разы. Система может использоваться как в промышленных, так и в учебных целях: начиная от изучения программирования в школах, где ученики могут изучать элементарные алгоритмы на наглядных примерах, продолжая олимпиадным программированием в университетах, где студенты могут готовиться к соревнованиям, оттачивая своё алгоритмическое мастерство, и заканчивая профессиональными разработками сложных алгоритмических систем, где производительность, точность и надежность алгоритмов играют одну из важнейших ролей.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Fast and high-quality modern software testing framework [Електронний ресурс]. – Режим доступу: URL: https://www.hpe.com/h20195/V2/GetPDF.aspx/4AA6-5193ENW.pdf. - Hewlett Packard software testing framework

2. The ACM-ICPC International Collegiate Programming Contest [Електронний ресурс]. – Режим доступу: URL: https://icpc.baylor.edu/help/about-icpc. – About ICPC

3. R.L. Graham. An efficient algorithm for determining the convex hull of a finite planar set [Текст] / L.R.Graham, Inform. Process. Lett. – 1972, С. 132-133