ПОДХОД К АВТОМАТИЗАЦИИ РАСПРЕДЕЛЕННОГО ТЕСТИРОВАНИЯ ГРАФИЧЕСКИХ ПРИЛОЖЕНИЙ

В.Ю. Коваленко, Д.А. Костюк, А.Г. Кричко
Кафедра ЭВМ и систем, Брестский государственный технический университет
Брест, Республика Беларусь
E-mail: dmitrykostiuk@gmail.com

Рассмотрен подход к параллельному тестированию графических приложений на разнотипных программно-аппаратных конфигурациях и платформах, предполагающий автоматическую систематизацию видеозаписей процесса работы программы средствами компьютерного зрения. Демонстрируется экспериментальная реализация предложенного подхода с использованием библиотеки OpenCV.

Проблему тестирования программного кода в условиях сильной фрагментации целевых аппаратных платформ нельзя назвать новой. По-настоящему заметной она стала с появлением легальных клонов IBM PC, благодаря их непредсказуемо разнящимся аппаратным и программным конфигурациям. Задача сохранила свою актуальность на платформе Wintel до настоящего момента, и даже увеличила ее с появлением существенно фрагментированной мобильной платформы Google Android. Многообразие условий, в которых должен быть работоспособным создаваемый программный продукт, может быть сколько-нибудь ощутимо покрыто тестами только в случае крупной компании-разработчика, причем в значительной степени задача решается за счет привлечения ресурсов, т. е. «в лоб».

Сложность увеличивается многократно, когда программный продукт представляет собой графическое приложение, рассчитанное на взаимодействие с пользователем [1, 2]. Хотя сегодня существует некоторое количество фреймворков для автоматического тестирования графических интерфейсов, они в большей степени являются средствами автоматического управления интерфейсом, чем инструментом для автоматизации анализа. А в случае тестирования эргономики и отслеживания рабочего процесса операторов основным инструментом и вовсе остается видеопротоколирование с последующим ручным анализом скринкастов (видео-потоков, записанных с экрана в ходе тестирования программы).

Последнее обстоятельство делает малоэффективной аренду мощностей для параллельного тестирования приложений, т. е. делегирование сторонним исполнителям массового запуска и выполнения тестируемого приложения. Если сбор статистики, генерируемой программой, выполняющейся в автоматическом режиме на разных конфигурациях программных и аппаратных средств, вполне осуществим и может быть выполнен в вычислительном облаке, то для графического интерфейса остается неподъемным вопрос анализ массива видеозаписей.

На рис. 1 представлена обобщенная структура распределенной системы тестирования, в рамках которой разработчик может привлекать сторонние вычислительные мощности для выполнения тестов при условии сколько-нибудь успешного решения проблемы автоматического анализа собранных данных.

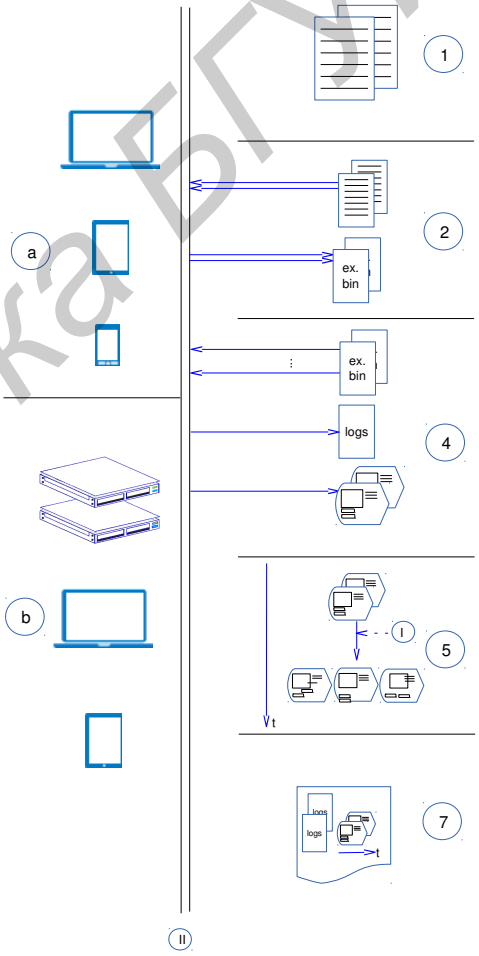

Рис. 1 – Схема распределенного тестирования приложений: I – действия тестировщика, II – система распределения задач, a – устройства разработчика, b – облачные устройства

Можно предложить использование следующего комплекта инструментов под свободными лицензиями, позволяющих организовать работу по приведенной схеме. Пропорциональное разделение задач и организацию клиент-серверной сборки кода может выполнять компилятор distcc, представляющий собой обертку для

gcc, установленных на клиентских машинах, при наличии сервера, организующего разделение, отсылку модулей исходного кода и сборку на своей стороне. Распределение исполняемых файлов на тестовые машины и контролируемое выполнение может осуществляться платформой BOINC [3]. Задача же распределенного тестирования графических интерфейсов до сих пор не решена.

Нами предложен и опробован подход к автоматической систематизации скринкастов, захваченных на параллельно выполняющихся тестовых приложениях, за счет системы компьютерного зрения. Для практической проверки работоспособности предложенного подхода была использована библиотека OpenCV.

В схеме тестирования применительно к графическому приложению могут быть выделены следующие стадии [2] (в соответствии с нумерацией на рис. 1): создание очередной версии исходного кода (1); опциональная рассылка заданий для раздельной компиляции и линковки, получение на выходе исполняемых модулей, файлов ресурсов и т.д. (2); пересылка исполняемых модулей и их параллельный запуск на тестовых платформах, обратная отсылка журнала работы и скринкастов (3); параллельный анализ изображений (4); формирование текстового и графического представления результатов (5).

Экземпляры тестируемой программы запускаются на различных платформах и конфигурациях. При этом снимаются скринкасты, отражающие изменения в интерфейсе, вызванные фреймворком автоматического тестирования либо действиями тестировщика. Анализ изображений производится по кадрам – они сравниваются друг с другом и, возможно, с заранее заданным эталоном. Для нивелирования различий во временной шкале потоков и ее неравномерности, сравнение кадров производится не последовательно, а по ключевым точкам – моментам, когда на разных машинах отображается определенное состояние интерфейса. По результатам создается граф с разветвлениями, соответствующими расхождению скринкастов, нормированных к единой временной шкале, и схождениями в точках, где различия исчезают [1]. Т. о. граф и отметки времени на нем дополнительно показывают различия в реакциях, ошибки и др. дополнительные данные.

В ходе апробации предложенный подход продемонстрировал свою работоспособность (см. граф для тестового приложения, приведенный на рис. 2). Реализация подхода была упрощена использованием свободно-распространяемых инструментов – в частности, метод SURF, реализованный в библиотеке OpenCV, для поиска особых точек изображения и создания их дескрипторов, инвариантных к масштабу и вращению (рис. 2-a), а также пакета Graphviz для визуализации результирующего графа (рис. 2-b).

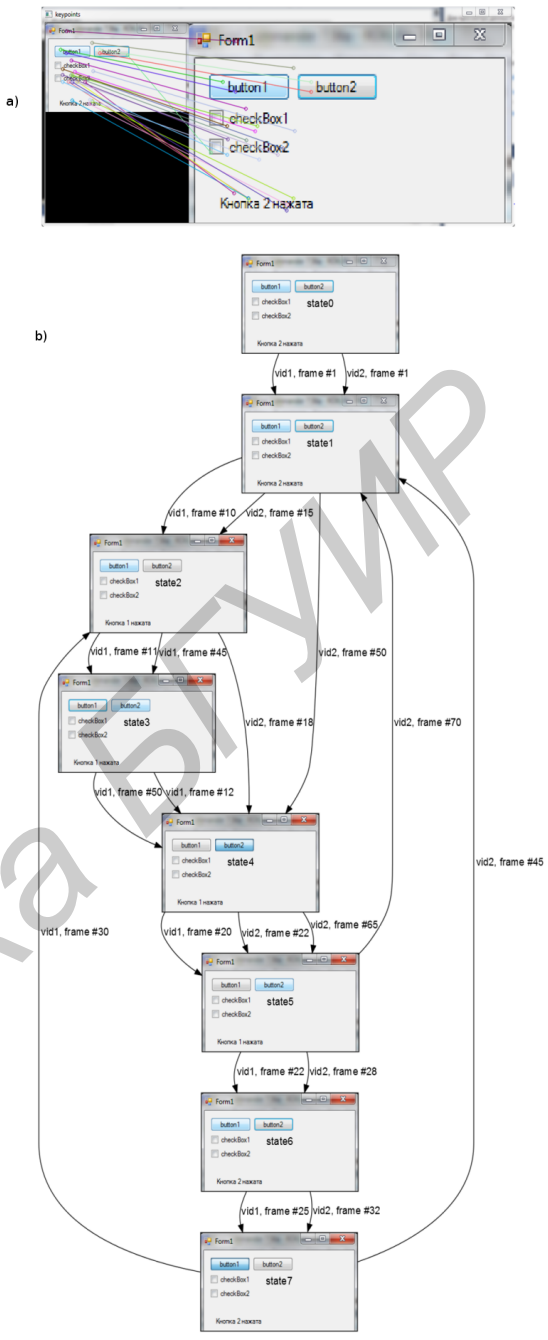

Рис. 2 – Пример автоматического выделения контрольных точек на кадрах (a) и результирующий граф (b)

1. Баранцев А. В. Генерация оптимизированных для ручного выполнения сценариев тестирования приложений с графическим интерфейсом пользователя [Электронный ресурс] / А. В. Баранцев, С. В. Грошев, В. А. Омельченко // Труды Института системного программирования РАН. 2009 – Режим доступа: http://citforum.ru/SE/testing/gui_test/ – Дата доступа: 14.09.2014.
2. Kostiuk, D. A. Approach to evaluate effectiveness of human-computer interaction with contemporary GUI / D. A. Kostiuk et al. // Третя міжнародна НПК FOSS Lviv 2013: Збірник науковых праць – Львів: ЛНУ ім. І. Франка, 2013. – С. 85–87.
3. Swierczewski, L. BOINC – Not only calculations / L. Swierczewski // Открытые технологии: матер. X Междунар. конф. Linux Vacation / Eastern Europe 2014 – Брест: «Альтернатива», 2014. – С. 123–125.