УДК 004.4, 004.9

Е. Н. Боженкова, Д. В. Иртегов, А. В. Киров, Т. В. Нестеренко, Т. Г. Чурина

Новосибирский государственный университет
ул. Пирогова, 2, Новосибирск, 630090, Россия

Институт систем информатики им. А. П. Ершова СО РАН
пр. Акад. Лаврентьева, 6, Новосибирск, 630090, Россия

E-mail: bozhenko@iis.nsk.su; dmitry.irtegov@gmail.com;
alkirpro@gmail.com; nest@iis.nsk.su; tanch@iis.nsk.su

АВТОМАТИЗИРОВАННАЯ СИСТЕМА ТЕСТИРОВАНИЯ NSUTS: ТРЕБОВАНИЯ И РАЗРАБОТКА ПРОТОТИПА*

В статье рассматриваются требования к построению автоматизированной системы тестирования, описывается прототип такой системы, предназначенной для проведения олимпиад по программированию и обеспечению процесса подготовки студентов и школьников на занятиях по программированию. Особое внимание уделяется обеспечению безопасности при передаче данных.

Ключевые слова: автоматизированная система тестирования, олимпиады по программированию, защита от мошенничества, изолирующая среда, NSUts.

Введение

Интенсивное развитие информационных технологий обуславливает необходимость постоянного совершенствования структуры и содержания образования для повышения качества подготовки ИТ-специалистов. Активно используемым методом оценки качества образования является применение автоматизированного тестирования знаний и навыков. Этот метод в различных формах применяется в школьном, высшем и послевузовском обучении. Например, сертификаты Microsoft Certified System Engineer, Cisco Engineer, Certified Lotus Professional выдаются на основе автоматизированного тестирования.

По сравнению с традиционными формами контроля автоматизированное тестирование требует меньших трудозатрат от учеников и преподавателей. Поэтому оно может использоваться для более интенсивного контроля над качеством обучения, чем это возможно в рамках традиционных форм. Оно удачно дополняет традиционные формы контроля и способствует повышению качества образования при незначительном росте его себестоимости. Также автоматизированное тестирование может использоваться учащимися для самостоятельного обучения и самоконтроля, в том числе при дистанционном обучении.

Эффективным комплексным средством проверки знаний и навыков программиста является написание программы, соответствующей заданным требованиям, с последующим ее тестированием. Поэтому программирование является привлекательной областью применения для автоматизированных систем тестирования. Такие системы отличаются от традиционных автоматизированных систем тестирования тем, что они предполагают написание испытуемыми программ, последующий запуск которых осуществляется на заранее подготовленном наборе тестов.

* Работа выполнена при проведения НИР в рамках реализации ФЦП «Научные и научно-педагогические кадры инновационной России» на 2009–2013 годы, ГК № П262 от 23.07.2009.

Эта система и методики могли бы решить целый ряд задач, связанных с совершенствованием качества образования в области информатики и закрепления молодежи в сфере науки, образования и высоких технологий. Такая система может использоваться для:

– промежуточного тестирования знаний и навыков по информатике и программированию студентов высших и среднетехнических учебных заведений;

– послевузовского, в том числе дистанционного, образования;

– организации уроков по информатике и кружков по программированию для школьников;

– проведения олимпиад по информатике и программированию всех уровней, в том числе, открытых интернет-олимпиад;

– профориентационного тестирования;

– тестирования в службе занятости и кадровых агентствах при отборе на вакансии, требующие навыков программирования;

– одного из этапов собеседования в ИТ-компаниях.

Вопрос о применимости системы или подходов, на которых основана такая система, для тестирования знаний и навыков в областях, не связанных непосредственно с программированием, требует отдельного исследования.

Требования, предъявляемые к автоматической системе тестирования знаний

С учетом десятилетнего опыта проведения олимпиад по программированию в Новосибирском государственном университете были сформулированы требования, которым должна удовлетворять система автоматизированного тестирования, пригодная для массового использования.

1. Система долцжна обеспечивать глубокое и адекватное тестирование знаний и навыков.

2. Система должна иметь эффективную защиту от мошенничества.

3. Система, или ее периферийные узлы, устанавливаемые в организациях, где проводится тестирование, должна быть проста в эксплуатации для специалистов средней квалификации.

Все три проблемы необходимо решать на двух уровнях: при разработке самой системы и при ее развертывании и эксплуатации.

Задача обеспечения глубины и адекватности тестирования должна решаться на этапе наполнения системы во время разработки тестов. Разработка новых заданий и их ввод в систему должны производиться на протяжении всего времени эксплуатации системы. Наличие банка опубликованных задач облегчает эту работу.

Задачи в банке должны быть проиндексированы по сложности и по набору знаний и навыков, необходимых для их решения. Такой банк можно использовать для создания наборов тестов заданного уровня сложности на заданные темы для контроля знаний и навыков в ходе учебного процесса при различных формах и уровнях обучения.

Для определения реальной сложности задачи ее желательно протестировать на практике, например, предложив на олимпиаде по информатике и программированию. Одна из возможных форм жизненного цикла задачи может выглядеть так: задача предварительно оценивается жюри, используется при проведении олимпиады, включается в банк вместе с наборами входных данных и решениями жюри, если они опубликованы.

Практика показывает, что олимпиады по информатике и программированию являются источниками качественных и интересных задач. Однако, как правило, эти задачи бывают достаточно сложны, а для промежуточного тестирования знаний в ходе учебного процесса могут требоваться более простые задачи. Поэтому олимпиады являются важным, но не единственным источником задач для наполнения банка.

Систематизация опубликованных задач, предлагавшихся на российских школьных олимпиадах по информатике, российских четвертьфиналах студенческой международной олимпиады ICPC (International Collegial Programming Contest), проводимой международной организацией ACM (Association for Computing Machinery) [1], и Открытой Всесибирской олимпиаде

[1] Сайт международной олимпиады по программированию ACM ICPC – http://cm.baylor.edu/welcome.icpc.

по программированию имени И. В. Поттосина [2] [1] позволит создать банк объемом более тысячи задач.

Важным требованием является требование защиты системы от мошенничества, так как при использовании традиционных форм автоматизированного тестирования основной проблемой является использование внешних источников информации, вплоть до списков правильных ответов. Если во время прохождения теста доступен Интернет, то информация, необходимая для ответов на большинство тестов школьного и вузовского уровня, может быть легко найдена. При широком применении какой-то определенной системы тестов можно ожидать, что в Интернет вскоре будет выложен полный набор правильных ответов на все задачи. В сочетании с современными средствами поиска решить эту проблему одним только расширением банка задач невозможно.

Описанная форма мошенничества может быть предотвращена регламентом проведения тестов. Как и при традиционных формах тестирования, доступ в Интернет и возможности использования внешних источников информации должны быть исключены. Регламент прохождения автоматического тестирования должен разрабатываться по аналогии с регламентами традиционных тестов, как автоматизированных, так и неавтоматизированных. В ситуациях, когда система используется для самотестирования учащихся, это требование может быть ослаблено или вообще снято.

Предлагаемая система тестирования основана на разработке испытуемыми программного кода и последующем запуске этого кода системой. При этом возникает возможность специфического типа мошенничества, отсутствующая при традиционном автоматическом тестировании, а именно: испытуемый вместо решения задачи может написать программу, которая обманывает тестирующую систему. Особенно серьезную проблему эта возможность представляет для широко внедряемой системы. У взломщика будет возможность исследовать эту систему и найти в ней слабые места, а затем технология взлома может быть легко растиражирована.

Важно понимать, что этот тип мошенничества опасен и для систем самотестирования. Сервер, на котором школьники или студенты проходят самотестирование, может быть взломан кем-то из учащихся просто для развлечения.

Обеспечение технической невозможности такого мошенничества представляет собой сложную задачу, решение которой во многом зависит от используемых технологий и языков программирования. На практике системы тестирования, применяемые при проверке школьных и вузовских олимпиад по программированию, не гарантируют технической невозможности мошенничества. Разработчики таких систем предполагают, что жюри хотя бы выборочно просматривает решения задач и наблюдает за работой тестового комплекса, а условия олимпиады предполагают дисквалификацию за попытки мошенничества. Разумеется, такой просмотр может быть эффективен, только если он будет осуществляться высококвалифицированным программистом, который может распознать попытку мошенничества даже при беглом просмотре кода. Поэтому такие системы следует называть системами механизированного, а не автоматизированного тестирования.

«Легкая» среда изолированного запуска тестируемых программ

Одно из направлений исследований средств технической борьбы с мошенничеством состоит в разработке «легкой» среды изолированного запуска тестируемых программ, использующей стандартные средства управления доступом уровня операционной системы, возможно с включением модулей ядра, блокирующих отдельные системные вызовы или группы системных вызовов для тестируемой программы [2]. Создание надежной среды такого типа позволит разработать автономную систему тестирования, которая может быть развернута непосредственно в том месте, где проходит тестирование. В Новосибирском государственном университете (НГУ) уже есть реализация изолирующей среды такого типа, но опыт ее использования еще недостаточен для утверждения, что он обеспечивает действительно надеж-

[2] Открытая Всесибирская олимпиада по программированию им. И. В. Поттосина – http://olimpic.nsu.ru/.

ную изоляцию, достаточную для длительной эксплуатации под присмотром специалистов средней квалификации.

Разработанная в НГУ изолирующая среда представляет собой часть комплекса тестирования олимпиад NSUts, а также может использоваться совместно с другими тестирующими системами. Изолирующая среда не допускает использования потенциально опасных функций для работы с операционной системой на уровне средств Win32 API. Она контролирует состояние запущенного приложения в момент исполнения и ресурсные ограничения. При попытке выйти за порог ограничений происходит немедленное завершение работы этого приложения. Также изолирующая среда обнаруживает исключения и ошибки выполнения. После завершения работы приложения изолирующая среда приходит в исходное состояние.

На настоящий момент реализована функциональность изолирующей среды, которая позволяет:

- *принимать прикладные программы, написанные участниками олимпиады*. Изолирующая среда представляет собой клиентское приложение (тестирующий клиент) к Web-серверу проведения олимпиады, который распределяет решения участников по клиентам. Тестирующий клиент по сети принимает программный код решения участника и параметры ограничения среды;
- *собирать программу соответствующим компилятором с необходимыми компиляционными ключами*. Перед запуском полученный программный код необходимо превратить в рабочее приложение. Для этого он собирается компилятором, определенным в параметрах переданных тестирующему клиенту. Контроль над ресурсами начинается на стадии компиляции. Это необходимо, так как код прикладной программы может содержать вложенные и циклические переопределения, которые, в свою очередь, могут привести к бесконечному времени компиляции. В системе используются специальные версии компиляторов. Из них исключены потенциально опасные библиотеки для работы с операционной системой на низком уровне. Это позволяет фактически закрыть присланному программному коду доступ к изменению настроек и состояния операционной системы. Сама компиляция запускается в упрощенной версии изоляции. Поэтому если присланная программа будет содержать ошибки, связанные со статическим выделением памяти огромных размеров, то она не скомпилируется, а следовательно, не сможет нанести вреда изолирующей среде. Также создана специальная учетная запись пользователя, у которой есть доступ только к компиляторам. Для обеспечения безопасности компиляция запускается от ее имени;
- *обеспечить среду запуска приложения*. Все тестируемые приложения должны находиться в одинаковых условиях. Для запуска собранной программы создана специальная учетная запись пользователя. Она имеет только одну рабочую директорию с монопольными правами доступа. В эту директорию складываются входные тесты, полученные от сервера тестирования. На нее, для обеспечения безопасности, средствами администрирования Windows устанавливается дисковая квота [3]. Все остальные директории для этого пользователя предназначены только для просмотра. Эта учетная запись пользователя не имеет прав на изменение параметров среды, поэтому от ее имени проводится запуск приложения. Таким образом, приложение не имеет возможности читать файлы, а тем более писать в файлы из любой другой директории, кроме своей. Это обеспечивает дисковую изоляцию, т. е. файловый ввод / вывод замкнут в пределах одной директории;
- *запускать прикладные приложения с заданными параметрами*. Входные данные передаются приложению в виде файла. Поэтому соответствующий файл помещается в рабочую директорию до запуска приложения. Кроме этого, для программ, написанных для JIT-компиляторов (Java), запускается среда исполнения (Java машина);
- *следить за ресурсными ограничениями, обеспечивая среду исполнения.* После запуска приложения изолирующая среда средствами WinApi контролирует ограничения среды исполнения, а именно процессорное время, память и общее время работы программы. Контроль общего времени нужен для того, чтобы программа не могла уйти в бесконечное ожидание при вызове блокирующих системных функций. Это очень важно, так как блокирующие функции на олимпиадах по программированию используются довольно часто.

[3] Безопасность в Windows XP – http://www.r-i-p.info/artview.php?id=642.

Если приложение попытается выйти за порог ограничений, то среда немедленно завершит его работу и оповестит об этом сервер;

- *после завершения исполнения привести среду в исходное состояние*. Каждое приложение должно быть запущенно с одинаковыми параметрами среды. Для этого изолирующая среда возвращает все свои параметры в исходное состояние: очищает память приложения, в том числе и его рабочую директорию, освобождает ресурсы, занимаемые приложением. После этого изолирующая среда готова к новому циклу жизни и тестированию новых программ [3].

«Тяжелая» среда изолированного запуска тестируемых программ

Запуск тестируемых приложений из-под пользователя с ограниченными правами обеспечивает довольно высокий уровень изоляции, но теоретически такая изоляция все-таки проницаема. В коде большинства операционных систем время от времени находят уязвимости, позволяющие пользовательскому процессу, а иногда и процессу, исполняющемуся с ограниченными привилегиями, получить полномочия локального администратора. Разработчики ОС, как правило, быстро выпускают обновления, позволяющие перекрыть такие уязвимости, но интервал от обнаружения такой уязвимости до ее исправления измеряется неделями; кроме того, не всегда все выпущенные производителем ОС обновления своевременно устанавливаются на тестирующий комплекс. Атаки такого рода (*zero day attack*) редки и обычно требуют от взломщика высокой квалификации, но представляют очень большую опасность. Так, именно через подбную уязвимость, была взломана корпоративная сеть компании *Google* в декабре 2009 г. [4]

Высокая степень изоляции, обеспечивающая эффективную защиту даже от большинства *zero day attack*, может быть достигнута запуском проверяемого кода в виртуальной машине. Для этого могут использоваться как традиционные виртуальные машины, так и контейнерная виртуализация, например, Parallels Virtuozzo Containers for Windows [5].

Однако попытка реализовать изолирующую среду на основе виртуальных машин наталкивается на слишком большое время запуска операционной системы Windows. Время нормального запуска Windows, как в традиционной виртуальной машине, так и в контейнере Virtuozzo, измеряется несколькими десятками секунд, а при некоторых комбинациях параметров даже минутами. Для сравнения, типичный лимит времени на один запуск тестируемой программы в соревнованиях по правилам ACM составляет от одной до десяти секунд, а большинство запусков на большинстве тестов завершается за время, измеряемое долями секунды. Таким образом, чтобы время запуска изолирующей среды стало сравнимо со временем работы тестирующей программы, время запуска должно быть уменьшено на два десятичных порядка.

В настоящее время удалось достичь определенных успехов в сокращении времени запуска виртуальной машины посредством создания образа виртуальной машины VirtualBox с заранее проинициализированной ОС, из которой удалены все неиспользуемые для проверки решений сервисы, в частности не инициализирована сетевая подсистема. Время приведения такого образа в рабочее состояние может быть доведено до нескольких секунд. Это все еще слишком много для того, чтобы переинициализировать среду для каждого запуска тестирующей программы. С целью оптимизации также можно рассмотреть варианты архитектуры тестирующего клиента, который использует одну копию изолирующей среды для нескольких последовательных запусков. Например, логично было бы делать запуск программы одного участника на разных наборах тестов в одной изолирующей среде. При этом программа участника была бы не идеально изолирована от самой себя, но программы разных участников были бы хорошо изолированы друг от друга.

---

[4] Google Hack Attack Was Ultra Sophisticated, New Details Show – http://www.wired.com/threatlevel/2010/01/operation-aurora/.

[5] http://www.parallels.com/ru/products/pvc45/.

Еще одна проблема, связанная с этим подходом, состоит в том, что по действующей с 2007 г. версии лицензионного соглашения Microsoft, каждая копия Windows, запускаемая в виртуальной машине, в том числе и в контейнере Virtuozzo, требует отдельной лицензии. В вузах, которые имеют подписку MSDN или другое соглашение массового лицензирования с достаточным запасом свободных лицензий, эта проблема может быть не очень существенной, но для школ она может оказаться серьезным ограничивающим фактором. Отметим также, что Parallels Virtuozzo Containers for Windows и некоторые другие среды виртуальных машин являются коммерческими программами. Если удастся достичь приемлемых параметров времени запуска только под одной из коммерческих виртуальных машин, то это могло бы ограничить применение изолирующей среды, основанной на данном подходе.

Прототип системы NSUTS

В НГУ создание автоматизированной системы тестирования началось с развитием олимпиад по программированию. Введение элементов коллективной игры и соревнований способствует заинтересованности студентов и интенсификации учебного процесса. В настоящее время в НГУ создан работающий прототип системы NSUTS, который апробируется при проведении школьных и студенческих олимпиад различного уровня.

Система состоит из двух основных подсистем: сервера олимпиад и тестирующего клиента. Сервер тестирования реализует логику проведения олимпиады и выполняет следующие функции.

1.Предоставляет веб-интерфейс для взаимодействия участников и членов жюри с системой тестирования.

2.Автоматизирует управление олимпиадой, осуществляет:

3.

- сбор и хранение условий задач, тестов и решений;
- обработку и отображение результатов тестирования;
- составление рейтинга мест, которые заняли участники соревнования;
- получение отчетов о проведении олимпиады;
- перетестирование решений участников;
- решение задач организационного плана, таких как регистрация участников, обеспечение обратной связи с жюри и других;
- администрирование олимпиад и туров.

Тестирующий клиент непосредственно осуществляет задачу тестирования решения. Решения участников, полученные сервером олимпиад, помещаются в очередь решений, откуда они забираются тестирующим клиентом. Тестирующий клиент в процессе обработки решения получает исходный код решения, информацию о необходимом для компиляции решения компиляторе, набор тестов. Исходный код компилируется и запускается в изолирующей среде на наборе тестов, результат работы программы сравнивается с эталонными результатами. Результат тестирования отправляется обратно на сервер, где происходит его обработка, составление рейтингов и т. д. Участники олимпиад и члены жюри взаимодействуют с системой исключительно через веб-интерфейс и не взаимодействуют с тестирующим клиентом напрямую.

Сервер олимпиад написан на языке Perl с использованием модулей архива CPAN и некоторых утилит командной оболочки (shell для Linux или cmd.exe для windows). Работа сервера олимпиад осуществляется под управлением веб-сервера apache2 [3]. Хранение данных приложения осуществляется в базе данных MySQL. Сервер тестирования может быть запущен как в ОС Linux, так и в ОС Windows.

Тестирующий клиент также написан на языке Perl с использованием командной оболочки MS Windows cmd.exe посредством использования bat-файлов. Отдельные компоненты реализованы на языке C с использованием WinAPI. Тестирующий клиент работает исключительно

под операционной системой MS Windows, что связано с требованием тестирования решений участников с использованием Windows-версий компиляторов [2].

Система проста при установке и эксплуатации и реализует следующие возможности.

1. Тестирующий модуль замеряет и ограничивает процессорное время и объем памяти у запущенного приложения, поскольку регламентом олимпиад устанавливаются ограничения на время выполнения программы и максимальный объем занимаемой памяти.
2. Тестирующий модуль обеспечивает безопасность запуска, запрещает изменять реестр Windows, ограничивает доступ к системным файлам и директориям и т. д.
3. Решения участников не имеют возможности использовать какие бы то ни было входные данные и параметры, кроме предписанных в условии задачи. В частности, исключается возможность доступа тестируемого решения к файлам с правильными ответами.
4. Время работы решения на одном тесте ограничено несколькими секундами. В последние годы у большинства задач лимит времени на прохождение одного теста составляет 1–2 секунды и редко превышает 10 секунд. Время запуска модуля тестирования должно быть того же порядка величины или меньше, так как при проведении крупных соревнований по программированию тестирующий клиент должен проверить до 5000 решений участников олимпиады на 50–100 тестах – это примерно полмиллиона запусков. В связи с этим время старта тестирующего модуля является критическим параметром, поэтому в настоящий момент система тестирования NSUTS использует принцип «легкой» изоляции. Таким образом, время запуска тестирующего модуля NSUTS не превышает времени работы тестируемой программы.

Заключение

Автоматизированная система тестирования учебных и тестовых заданий по программированию и методики применения системы для проверки различных навыков, связанных с программированием, полезны при изучении основ программирования, алгоритмов и структур данных, работы с базами данных, математического моделирования, олимпиадных задач.

Для обеспечения работы олимпиад по информатике и программированию была поставлена цель разработать систему, поддерживающую проверку правильности решений задач, которые представляются программой на каком-либо языке программирования. Такая система должна быть простой в установке, настройке и использовании без необходимости привлечения программиста.

В статье представлены требования, предъявляемые к автоматизированной системе тестирования, и описаны способы их реализации. На их основе в НГУ создан работающий прототип системы NSUts, который апробируется при проведении школьных и студенческих олимпиад различного уровня.

Система тестирования должна сопровождаться созданием банка задач по программированию, предназначенных для автоматизированного тестирования, на основе задач, предлагавшихся на школьных и вузовских олимпиадах по программированию в России и за рубежом.

Обе эти задачи реализуют потребность в создании виртуальной информационно-образовательной среды, включающей базу олимпиадных задач разных лет, систему автоматического тестирования и проверки знаний, базу данных, позволяющую получать информацию о победителях и участниках олимпиад по их принадлежности к вузам и субъектам РФ, проводить статистические срезы регионов по уровню подготовки студентов и школьников.

Список литературы

1. *Чурина Т. Г., Боженкова Е. Н., Нестеренко Т. В.* Задачи Открытой Всесибирской олимпиады по программированию имени И. В. Поттосина: от теории к практике // Вестн. Новосиб. гос. ун-та. Серия: Информационные технологии. 2007. Т. 5, вып. 1. С. 40–46.

2. *Чурина Т. Г, Иртегов Д. В.* Требования к автоматической системе тестирования знаний // Тр. VI Междунар. конф. «Интеллектуальные технологии в образовании, экономике и управлении». Воронеж, 2009.

3. Apache HTTP Server Version 2.2 Documentation. URL: http://httpd.apache.org/docs/2.2/.

Материал поступил в редколлегию 24.09.2010

E. N. Bozhenkova, D. V. Irtegov, A. V. Kirov, T. V. Nesterenko, T. G. Churina

**AUTOMATED TESTING SYSTEM NSUTS:
REQUIREMENTS AND THE DEVELOPMENT OF PROTOTYPE**

The article deals with the requirements for the construction of automated testing system, and describes a prototype system for conducting programming contests, and ensure the preparation of students and pupils in classes on programming. Particular attention is paid to ensuring the security during data transmission.

Keywords: automated testing system, programming contests, fraud protection, insulating environment, NSUts.