часть шаблона. Кроме того, разработчик готовит набор тестовых примеров. Выполнение задания считается успешным ,если тестирование прошло успешно на всех тестовых обстановках для данного задания.

Описание практикума представляет собой xml-файл. При выполнении практикума учеником создаётся его личный файл («рабочая тетрадь»). Рабочий файл ссылается на файл курса, в нём хранятся все сведения о прохождении курса учеником.

В настоящее время завершается разработка специализированного редактора практикумов TaskEdit. Этот редактор работает отдельно от системы КуМир. Рекомендуется при создании практикума иметь одновременно открытые окно редактора практикумов, в котором создаётся практикум, и Кумира, в котором просматриваются внесённые в практикум изменения и редактируются шаблоны заданий и обстановки используемых исполнителей.

Главное окно редактора практикумов состоит из двух частей. В левой части окна производятся операции с «деревом заданий»: добавление новых заданий, удаление ненужных и изменение порядка заданий. Для удобства, поскольку задания часто бывают однотипными, добавление производится копированием существующего задания. В правой части окна производится редактирование текущего задания — изменение описания, выбор заготовки программы , исполнителей и обстановок к ним.

В. В. Яковлев, Д. В. Хачко, Т. Р. Джелядин, М. А. Ройтберг
Москва, Пущино, НИИСИ РАН

Проект: Кумир `http://niisi.ru/kumir`

Кумир 2.0: новые цели — новые решения

Аннотация

Система обучения программированию Кумир (Комплект Учебных МИРов) была разработана в конце 80-х годов, она была реализована для существующего на тот момент парка компьютеров. К 2005–2011 г. была разработана кроссплатформенная система программирования Кумир 1.x, в которой на новом уровне были воспроизведены все возможности исходного Кумир'а, а также ряд новых возможностей, в частности, практикумы и новые исполнители. В сообщении представлен проект новой системы Кумир 2.0 и первые результаты реализации

этого проекта. В системе Кумир 2.0 будут сохранены все возможности Кумир 1.x. Основные отличия КуМир 2.0 от Кумир 1.x с точки зрения пользователя — увеличение скорости выполнения на порядок, с точки зрения разработчика — модульная архитектура.

Постановка задачи

В ходе эксплуатации систем семейства Кумир 1.x стола понятна необходимость расширения возможностей системы. Основными из них являются следующие.

1. Возможность выполнения программ со скоростью, сопоставимой с промышленными языками программирования. Это требование обусловлено необходимостью решения ресурсоемких задач, как правило возникающих в сфере олимпиадного программирования. От Кумира требуется скорость работы если не уровня C/C++, то как минимум не хуже чем у Python, Perl или Ruby.
2. Возможность интеграции в системы автоматизированной проверки заданий на удалённом сервере, например, возможность работы в паре с системой ejudge[1].
3. Наличие системы поддержки практикумов с возможностью интеграции в существующие системы управления курсами, например Moodle[2].
4. Поддержка различных учебных языков программирования с использованием пользовательского интерфейса Кумир а также набора Кумир-исполнителей.
5. Возможность использования исполнителей системы[3] как самостоятельных учебных продуктов, не входящих в поставку Кумир.

Ускорение выполнения программ

Существующая ранее реализация Кумир 1.x во время «компиляции» выполняет анализ корректности программ, написанных на алгоритмическом языке[4], и строит упрощённое представление программы для выполнителя. Такой подход является крайне неэффективным с точки зрения скорости выполнения и сохраняется как рудимент с тех времён, когда перед нами не стояла задача реализации эффективной системы выполнения. Для ускорения выполнения необходимо как

минимум упростить выполнитель за счёт переноса части его функциональности в компилятор, а для достижения большей скорости выполнения — получать выполняемый машинный код.

Возможность работы без пользовательского интерфейса

В 2010 году для апробации проведения компьютерного ЕГЭ с использованием языка «Кумир»[5, 6], нами была создана консольная версия интерпретатора. Данный инструмент был создан под конкретную задачу как ответвление от кодовой базы Кумира путём выбрасывания кода, зависимого от библиотек пользовательскоко интерфейса. В новой версии необходимо изначально предусмотреть возможность работы как с использованием графического пользовательского интерфейса, так и возможность независимого использования отдельных компонент системы.

Интеграция системы поддержки курсов в существующую инфраструктуру

Во-первых, система поддержки практикумов должна иметь возможность работы с удалённым сервером Moodle[2], не используя сторонних программ, в частности – не требуя запущенного веб-браузера. Во-вторых, пользовательский интерфейс должен быть спроектирован таким образом, чтобы точкой входа пользователя в интерфейс системы был не текстовый редактор с пустой программой, а некоторая осмысленная сущность, например веб-страница со списком невыполненных заданий.

Интеграция сторонних компиляторов и интерпретаторов в систему Кумир

Опыт работы с системой показал эффективность нашего подхода к редактированию программ: проверка программы при переходе на новую строку и отображение сообщений на полях редактора. Кроме того, на данный момент для многих языков программирования, в частности Паскаль, не существует кросплатформенной современной среды разработки, пригодной для учебных целей. Мы ставим перед собой задачу восполнить данный пробел, создав тем самым полный единообразный комплект программного обеспечения для обучения

программированию в рамках школьной программы. Также представляется интересным возможность использования графических Кумир-исполнителей при написании программ на других учебных языках программирования.

Исполнители

Система Кумир, помимо реализации своего языка программирования, содержит набор исполнителей, которые могут быть использованы без программирования. Для этого у большинства исполнителей имеется «пульт управления», предназначенный для решения задач[3]. Естественным образом возникает потребность в исполнителях с пультами, как отдельных программных продуктов, предназначенных для младшей школы. Соответственно, новая система должна быть спроектирована таким образом, чтобы данные исполнители можно было использовать как отдельные программы, устанавливаемые на компьютер, а ещё лучше – как веб-приложения.

Реализация системы

Система Кумир 2.x состоит из набора *компонент*, между которыми могут быть зависимости, а также набора небольших программ – *стартёров*, которые загружают по некоторой предопределённой конфигурации «свой» набор компонент и передают управление стартовой компоненте. Данная архитектура реализована на основе системы PlugIn'ов библиотеки `Qt4`[7]. Компонентами системы являются анализатор языка программирования Кумир, генераторы выполнимого байт-кода и C-кода, различные части пользовательского интерфейса а также исполнители. Стартёрами являются консольный компилятор или консольный выполнитель байт-кода, а также различные варианты пользовательского интерфейса для разных областей применения.

Анализаторы программ

Анализатор выполняет проверку програм на корректность, предоставляет информацию и подсветке кода и расстановке отступов редактору, и, в случае языка «Кумир» – дерево разбора программы[8]. Помимо анализатора языка «Кумир», возможна поддержка других языков программирования. Так например, у нас имеется эксперимен-

тальная реализация анализатора для языка Паскаль, основанная на слегка модифицированном компиляторе `FreePascal`[9]. Модицикация компилятора была обусловленна во-первых необходимостью работать в режиме «запрос-ответ» без перезапуска компилятора, а во-вторых – необходимостью получения дополнительных сведений (подсветка синтаксиса) об анализируемой программе.

Генераторы кода

На данный момент у нас имеется два генератора кода, которые по ранее построенному дереву разбора Кумир-программы строят: а) выполнимый компактный байт-код (интерпретатор байт-кода является отдельной компонентой системы); б) C-программу для дальнейшей трансляции в машинный код средствами `gcc` или `clang`.

Использование байт-кода позволяет полностью контролировать процесс выполнения программы, в то время как выполнение программы, полученной через C – позволяет достигнуть максимальной скорости выполнения. В то же время, компиляция C-программы и запуск отдельного процесса требуют большего времени, поэтому использование данного способа генерации кода и выполнения оправдано только для ресурсоемких задач.

В настоящее время проводятся эксперименты по использованию библиотеки `LLVM`[10] для генерации кода. В случае, если получится создать надёжную кросс-платформенную реализацию контролируемого выполнения программы, то генератор `LLVM`-кода станет основным, а вместо использования C-кода для генерации машинного, будет использована технология JIT-компиляции[11].

Пользовательский интерфейс

Пользовательский интерфейс будет существовать как минимум в двух различных вариантах: *классический* – для средней школы, и *профессиональный* – для старшей школы и олимпиадного применения. Классический вариант интерфейса максимально идентичен таковому в существующей системе Кумир 1.x, в то время профессиональный вариант будет предоставлять пользователю доступ ко всем новым возможностям системы. Также возможно создание интерфейсов, ориентированных под конкретные задачи, например подготовку

к экзаменам, используя систему курсов[12], или сдачу экзамена, используя для проверки удалённый сервер.

Исполнители

На данный момент только отрабатывается технология для взаимодействия с исполнителями. Предполагается некоторую их часть реализовать в виде Web-приложений, отображаемых средствами `WebKit`. Кроме того, в планах обеспечить возможность работы с исполнителями используя другие языки программирования. Для этого планируется реализовать инструмент, генерирующий по формальному описанию исполнителя соответствующие оболочки для различных поддерживаемых языков программирования.

Текущее состояние и дальнейшие планы

Релиз системы 2.0 предполагается, когда будут полностью реализованы и протестированы, то есть успешно пройдут все тесты из системы тестирования Кумир 1.x, следующие компоненты:

1. анализатор языка Кумир — уже реализован и успешно проходит тесты;
2. генератор и выполнитель байткода — уже реализованы и успешно проходят тесты;
3. стандартная библиотека функций языка Кумир — реализовано и протестировано всё, кроме файлового и терминального ввода-вывода.

На данный момент также реализован пользовательский интерфейс (в различных вариантах), однако он требует доработки (в том числе концептуальной), поэтому появится только в следующем релизе 2.1.

Также в дальнейших релизах предполагается реализация:

1. Выполнение программ средствами `LLVM`.
2. Создание варианта интерфейса, ориентированного на работу с системой курсов.
3. Реализация различных исполнителей.
4. Поддержка языка программирования `Паскаль`, и возможно, `Python`.

5. Поддержка новых целевых платформ для языка Кумир: `Lego NXT` [3, 13] и `Arduino`[14].

Исходные тексты системы успешно собираются на любом современном дистрибутиве Linux и свободно распространяются под свободной лицензией GPL 2.

SVN-репозиторий исходных текстов находится по адресу [15].

Литература

[1] А. В. Чернов *Система тестирования ejudge.* Вторая конференция «Свободное програмное обеспечение в высшей школе». М.: AltLinux, 2007.

[2] `http://moodle.org/`

[3] А. Г. Кушниренко, А. Г. Леонов, М. А. Ройтберг, В. И. Хачко, Д. В. Хачко, В. В. Тарасова, В. В. Яковлев *Новые Миры в системе КуМир.* Пятая конференция «Свободное програмное обеспечение в высшей школе». М.: AltLinux, 2010. с. 56–58.

[4] *Информатика: 7–9 кл.: Учеб. для общеобразоват. учр.* А. Г. Кушниренко, Г. В. Лебедев, Я. Н. Зайдельман. М.: Дрофа, 2003. 335 с.

[5] А. Г. Кушниренко, А. Г. Леонов, В. Р. Лещинер, Д. А. Путимцев, П. С. Шестаков *Структура программного обеспечения для проведения ЕГЭ по информатике и ИКТ в компьютеризированной форме.* Шестая конференция «Свободное програмное обеспечение в высшей школе». М.: AltLinux, 2011. с. 21–23.

[6] А. Г. Кушниренко, А. Г. Леонов, М. А. Ройтберг *Принципы выбора языков и сред программирования при проведении ЕГЭ по информатике и ИКТ в компьютеризированной форме.* Шестая конференция «Свободное програмное обеспечение в высшей школе». М.: AltLinux, 2011. с. 24–26.

[7] `http://qt.nokia.com/`

[8] *Компиляторы. Принципы, технологии, инструментарий.* А. Ахо, Р. Сети, Дж. Ульман. М.: Вильямс, 2008. 768 с.

[9] `http://freepascal.org/`

[10] `http://llvm.org/`

[11] Hans Wennborg *Emulator Speed-up Using JIT and LLVM.* Master's Thesis, Lund University, Lund, Sweden, Jan. 2010.

[12] Д. В. Хачко, Д. П. Кириенко, А. Г. Кушниренко, А. Г. Леонов, М. А. Ройтберг *Поддержка курсов в системе КуМир.* Шестая конференция «Свободное програмное обеспечение в высшей школе». М.: AltLinux, 2011. с. 19–21.

[13] `http://mindstorms.lego.com/en-us/Default.aspx`

[14] `http://www.arduino.cc/`

[15] `http://lpm.org.ru/svn/kumir2/`

М. А. Ройтберг
Москва, НИИСИ РАН

Новые тенденции преподавания информатики в школе

Аннотация

На конференции «Свободное программное обеспечение в высшей школе — 2011» был представлен доклад об апробации компьютерного ЕГЭ, проведённой в конце 2010 г. За истёкший год в школьной информатике произошёл ряд интересных событий, в частности, была разработана программа по информатике для основной школы, соответствующая новым образовательным стандартам, произошли изменения в корпусе заданий ЕГЭ по информатике. В сообщении будет представлен обзор новых тенденций в развитии школьной информатики и о связи этих тенденций с высшей школой.

В докладе будет представлен анализ модели ЕГЭ по информатике 2012 г. Эта модель является преемственной по отношению к модели 2011 г., но имеет ряд важных отличий. Если говорить о тенденциях, то они состоят в следующем:

1. увеличение количества заданий, проверяющих умение ученика понимать представленный ему алгоритм, анализировать результаты выполнения алгоритма без выполнения полной трассировки;
2. снижение «арифметической нагрузки» на ученика, снижение риска случайных ощибок;
3. введение новых типов задач, в частности, связанных с анализом графов.