Электронная информационная образовательная среда вуза, информатизация деятельности образовательных организаций и информационная безопасность в образовании

УДК 004.414.2:[371.278:371.38]

Андреева Т. А.

АВТОМАТИЗИРОВАННАЯ ПОДГОТОВКА ЗАДАЧНЫХ КОМПЛЕКТОВ ДЛЯ ОЛИМПИАД ПО ПРОГРАММИРОВАНИЮ

Татьяна Анатольевна Андреева

м.н.с.

ata@iis.nsk.su

Федеральное государственное бюджетное учреждение науки

Институт систем информатики им. А.П. Ершова

Сибирского отделения Российской академии наук (ИСИ СО РАН),

г. Новосибирск

AUTOMATED CREATION OF THE PROBLEM COMPLEXES FOR PROGRAMMING CONTESTS

Tatiana A. Andreyeva

The A.P. Ershov Institute of Informatics Systems

(Siberian Branch of the Russian Academy of Sciences), Novosibirsk

Аннотация. *Системы автоматической проверки решений задач могут применяться не только на олимпиадах, но и в учебном процессе (особенно в вузах). Для того чтобы сделать их доступными более широкому кругу*

преподавателей, необходима автоматизация не только самого процесса проверки, но и процесса подготовки к этой проверке.

Предлагаемая статья рассматривает различные аспекты автоматического создания задачных комплектов (в частности, тестовых наборов) и автоматической проверки решений задач по программированию.

Abstract. *Systems for automated correctness-checking can be used both at programming contests and in the teaching process. To make these systems useful for a wider circle of school and university teachers, it is necessary to automate not only the checking process itself but the preliminary preparation of problem complexes as well.*

This article concerns various aspects of automated correctness-checking and of the automated creation of the problem complexes and test sets.

Ключевые слова: *автоматическая генерация тестов, анализ текстов на естественном языке.*

Keywords: *automated test generation, analysis of texts in a natural language.*

Введение

На сегодняшний день получили широкое распространение онлайн олимпиады всевозможных уровней и направлений: от предметов школьной программы (например, [6]) до профессиональных состязаний типа международных олимпиад ACM [7]. Все эти олимпиады преследуют не только проверочные, но и образовательные цели, поэтому желательно расширить их применение в учебном процессе.

Для того чтобы процессы подготовки и проведения «собственной» олимпиады стали доступны любому школьному учителю, необходимо их максимально автоматизировать. Использование системы автоматической подготовки олимпиадных задач также поможет специалистам более высокого

уровня минимизировать количество возможных ошибок и уменьшить временные затраты.

Отличительные свойства решений задач по программированию

Большинство предметных олимпиад организовано по типу онлайн тестирования: участник выбирает один или несколько правильных ответов из нескольких предложенных вариантов. Все правильные решения заранее известны. Понятно, что процессы подготовки и проверки подобных тестов легко автоматизируются.

Однако совсем иначе обстоит дело в случае олимпиад по программированию, и обусловлено это сущностью самого программирования как науки. Решение любой задачи по программированию является лишь отдельным представителем целого и, вообще говоря, бесконечного класса эквивалентных решений. Невозможно заранее перечислить все программы, входящие в искомый класс решений.

Таким образом, основной проблемой при проверке этих решений является выработка набора критериев, позволяющих с некоторой долей уверенности отнести проверяемое решение к классу верных.

Проверка по принципу «черного ящика»

Решением любой задачи по программированию является программа. При проверке её «правильности» (иными словами, её соответствия критериям, описывающим класс верных решений) можно использовать текстовый и синтаксический анализ текста программы, однако в условиях реального времени чаще всего ограничиваются тестированием полученного на её основе исполняемого модуля (по умолчанию считая ничтожной вероятность появления в нём дополнительных ошибок, привнесенных на этапе компиляции или оптимизации).

Таким образом, решение о «правильности» программы принимается лишь на основе результатов, получаемых при помощи её исполняемого модуля. Этот способ тестирования, когда заключение о некотором внутреннем

свойстве исследуемого объекта делается лишь на основании его откликов на различные раздражители, получил название «чёрного ящика» [4].

В случае задач по программированию объектом исследования является программа, «раздражителями» – входные данные, реакцией на них – выходные данные. Однако получает входные данные и выдает выходные не сама исследуемая программа, а её исполняемый модуль. Вопрос о тождественности их «правильности» в случае онлайн-олимпиад обходится через включение условия о едином для всех программ допустимом компиляторе (хотя это, разумеется, не является доказательством).

В дальнейшем мы также будем полагать, что программа «верна» тогда и только тогда, когда «верен» её исполняемый модуль. Иными словами, мы предполагаем, что во время компиляции в решении не появляются дополнительные ошибки.

Критерии правильности решений

Перейдём теперь к рассмотрению критериев, позволяющих обнаруживать «правильные» и «не правильные» решения. Поскольку вывод о «правильности» программы нужно сделать только на основании результатов обработки входных данных, то и критерии правильности необходимо сформулировать только с использованием этих данных.

Множество всех возможных наборов входных данных обычно является бесконечным. Однако его всегда можно разбить на непересекающиеся классы эквивалентности, где в один класс попадают наборы, которые «на выходе» дают одинаковые или однотипные (например, описываемые одним законом) результаты. Понятно, что один такой класс эквивалентности описывает один частный случай в решении исходной задачи.

Считается [4, 8], что для установления правильности проверяемого решения достаточно проверить его на конечном количестве представителей каждого класса эквивалентности. В крайнем случае из каждого класса берут лишь одного представителя.

Очевидно, что для надежности заключения о правильности проверяемой программы разбиение *множества допустимых входных данных* (МДВД) на классы эквивалентности имеет первостепенное значение.

Задача построения разбиения области входных данных на классы эквивалентности является непростой даже для специалистов [4]. Теория, позволяющая правильно выбирать представителей из каждого класса, также пока недостаточно проработана [5].

Поэтому, для того чтобы снизить количество ошибок, разработку тестовых наборов необходимо максимально автоматизировать. Возможно ли сделать ее полностью автоматической? К сожалению, пока нет. Однако разработать систему, помогающую минимизировать количество ошибок при подготовке к автоматической проверке решений, представляется вполне возможным [1].

Тестовые наборы

Тестом называется пара *входные данные – выходные данные*. В олимпиадной практике эти данные записываются в виде отдельных текстовых файлов (input.txt и output.txt), однако в других случаях могут храниться, например, в общей базе данных иного формата.

В каждом конкретном тесте входные данные – это представитель одного из классов эквивалентности, на которые разбито всё множество допустимых входных данных. А выходные данные – «правильный» ответ, который должен быть получен при «правильной» работе проверяемой программы.

Поскольку любая программа должна обладать свойством массовости, то решение о её правильности невозможно принять, исходя только из результатов выполнения одного теста. Поэтому для проверки необходимо использовать *тестовые наборы,* состоящие из нескольких тестов, причем среди тестов обязаны присутствовать представители всех классов эквивалентности.

Тестовые наборы – это наборы критериев, на основании которых делается вывод о «правильности» программы.

Процесс автоматической проверки решений

Существует большое количество *систем автоматической проверки решений* (АПР), однако все они работают по одному и тому же принципу.

Для каждого теста, входящего в тестовый набор, система АПР выполняет следующие действия:

1. Производит запуск проверяемого *исполняемого программного модуля* (ИПМ), подставляя в его входной поток данные, взятые из входного файла текущего теста.

2. Фиксирует ошибки, возникшие в процессе выполнения ИПМ и повлекшие преждевременную остановку работы программы (переполнение памяти, «деление на ноль», т.п.), а также следит за соблюдением временных ограничений, если таковые прописаны в условии задачи. При превышении лимита времени (его причинами могут быть ожидание ввода с неверного потока входных данных, зацикливание, неэффективный алгоритм, т.п.) выполнение ИПМ прерывается искусственно.

3. Если после регулярного или принудительного завершения работы ИПМ получен файл с выходными данными, то система АПР производит проверку полученного ответа: либо методом вычисления его допустимости, либо методом сравнения его на совпадение с одним или несколькими эталонными ответами. Выбор метода проверки зависит от типа решения (единственный ответ, один из нескольких верных ответов, все возможные верные ответы, попадание в заданный интервал).

4. Если ответ получен и принято решение о его правильности, система АПР рапортует об успешном прохождении теста.

Правильные решения

Заключение о правильности проверяемого решения выносится на основании проверки результатов исполнения всех тестов из тестового набора. Наиболее естественным показателем «правильности» является показатель тестового охвата. При условии, что имеющийся тестовый набор покрывает все множество допустимых входных и выходных данных, а также корректно

учитывает все возможные частные случаи, например, «более правильной» программой можно назвать ту, которая верно обрабатывает большее количество тестов из этого тестового набора.

На олимпиадах нижних уровней, а также при использовании системы АПР в учебном процессе подобная градация «правильности» имеет смысл, так как на её основе легко определить количество начисленных за решение задачи баллов. Однако на более серьёзных олимпиадах придерживаются строгой дихотомии «верно/не верно»: решение считается верным только в том случае, если его ИПМ правильно обработал все тесты из тестового набора [7].

Поэтому назовем *правильным* такое решение задачи, которое верно обрабатывает все возможные частные случаи, то есть все классы эквивалентных входных данных, на которые распадается область возможных входных данных. Таким образом, доказательство правильности решения требует его проверки на правильно выбранных представителях этих классов выходных данных.

Задачные комплекты

Для проведения автоматической проверки при помощи системы АПР, необходимо предварительно создать *задачный комплект*. В него входят:

1) условие задачи (текст);
2) спецификация форматов входных и выходных данных;
3) тестовый набор;
4) эталонное решение.

Ошибки могут встретиться при создании любой из этих четырех частей [3], но наиболее сложной является задача создания полного тестового набора, а особенно проверки и доказательства его полноты [2, 4, 5, 8].

Чем выше уровень олимпиады, тем большее количество человек принимают участие в создании задачного комплекта. Разумеется, один человек может быть одновременно автором и условия, и тестов, и эталонного реше-

ния. Однако для повышения надежности работу над задачным комплектом целесообразно поручать нескольким специалистам.

При индивидуальной разработке задачного комплекта его автор несет ответственность за все возможные ошибки. Чтобы облегчить самопроверку, большую часть работы необходимо передать автоматизированной *системе подготовки задачных наборов* (СПЗН), которая поможет автору(-ам) задачи в поиске ошибок, неточностей, неполноты и т.п.

Создание задачного комплекта

Процесс создания задачного комплекта можно разбить на три этапа. Он является итеративным: по результатам работы над каждым из этапов может возникнуть необходимость внести уточнения или исправления в любой из двух остальных.

1. Исходя из первоначальной идеи задачи, автор условия ставит ограничения R на используемые переменные и делает первоначальное разбиение D множества входных/выходных данных, обеспечивая учет по возможности большего количества частных случаев. Он же предварительно задает спецификации S входных и выходных данных.

2. Опираясь на спецификации S и разбиение D, проектируется первоначальный тестовый набор, который используется при отладке эталонного решения.

3. Создается и отлаживается *эталонное решение* (ЭР) задачи.

Видно, что от ошибок не может быть застрахован ни один этап разработки [3]. Неправильное разбиение пространства данных повлечет за собой неполноту тестового набора, что, в свою очередь, отразится на правильности эталонного решения.

Процесс создания задачного комплекта является итеративным. По результатам работы над каждым из этих этапов может возникнуть необходимость внести уточнения или исправления в любой из двух остальных.

Создание тестовых наборов

Процессы создания эталонного решения и тестового набора тесно переплетены: для того чтобы создать правильный тестовый набор, необходимо правильное эталонное решение, а для отладки эталонного решения требуется полный тестовый набор. Полнота тестового набора (полнота покрытия его тестами всех классов эквивалентности, на которые разбивается пространство возможных решений) является существенным условием для определения правильности проверяемого решения (в т.ч. и эталонного).

Зачастую автором эталонного решения и тестового набора является один и тот же человек (преподаватель, организатор олимпиады, т.п.), причём некоторые тесты из «рабочего» тестового набора, использованного при отладке эталонного решения, затем входят и в итоговый тестовый набор. Очевидно, что если в эталонном решении имеются ошибки, которые не были обнаружены при помощи авторского тестового набора, то эти же ошибки не будут обнаружены и в других программах уже во время работы системы АПР.

Наличие возможности автоматически создавать и проверять тестовые наборы повысит надежность эталонного решения, а в некоторых случаях может также помочь заметить и исправить неполноту тестового набора.

Генерирование и проверка тестовых наборов

Жестко заданные спецификации форматов входных и выходных данных позволяют создавать тестовые наборы автоматически.

Основной особенностью тестирования олимпиадных задач является использование только допустимых входных данных (отвечающих всем поставленным в условии задачи ограничениям). Таким образом, подлежащая покрытию область данных заметно сужается.

Если имеется разбиение множества допустимых входных данных (МДВД) на классы эквивалентности, то формирование тестов может производиться автоматически с применением рандомизированных процедур выбора, по комбинаторному принципу в сочетании с принципом наибольшего

охвата данных. Более подробно вопросы выбора представителей классов эквивалентности будут рассмотрены в отдельной статье.

При адекватном учете всех классов эквивалентности и граничных условий (напомним, что особые точки из рассмотрения заведомо исключаются), полученный тестовый набор будет максимально полно покрывать МДВД. Доказательство полноты покрытия в каждом конкретном случае остаётся «на совести» человека.

Поскольку помимо автоматически сгенерированных тестов в тестовый набор могут быть включены также и авторские тесты, созданные вручную, необходимо осуществлять проверку всего тестового набора на соответствие исходным спецификациям.

К сожалению, в большинстве случаев разбиение МДВД на классы эквивалентности невозможно «извлечь» из спецификаций форматов входных и выходных данных. Обычно это разбиение вытекает не из формы спецификаций, а из способа решения задачи, который является той самой «неизвестной», которую нужно найти. Собственно поиск такого разбиения (иными словами, учет всех возможных частных случаев) и является основной трудностью при решении задач, и автоматизировать его пока не удается.

Таким образом, автоматизация процесса создания тестов на этапе генерации входных данных может быть лишь частичной.

Генератором сложных выходных данных обычно служит программа, являющаяся авторским решением исходной задачи, простые данные часто создают вручную. В обоих вариантах возможны ошибки формата ввода-вывода. Поэтому необходимо осуществлять проверку всех выходных данных (вне зависимости от способа их получения) на соответствие указанным в условии задачи спецификациям. Этот этап также легко поддается автоматизации.

Извлечение спецификаций из текста

Как видим, для генерирования и проверки входных и выходных данных важную роль играют спецификации формата, которые описываются в тексте

задачи. Сюда же относятся описания ограничений и условий, которым должны удовлетворять решения. Для извлечения спецификаций из текста задачи необходим текстовый анализ, однако его результаты всё равно подлежат проверке человеком, поэтому желательно, чтобы «внутренний» вид спецификаций, используемый генератором тестовых наборов, не слишком отличался от их «внешнего» вида, предназначенного для людей.

Представляется, что для частного случая олимпиадных задач автоматическое извлечение спецификаций не составит проблемы, поскольку для них давно уже выработаны и стали общепринятыми правила описания форматов входных и выходных данных.

Система автоматизированной подготовки задачных наборов

Сведём воедино требования к функционалу *автоматизированной подготовки задачных наборов* (АПЗН).

На первом этапе разработки задачного комплекта АПЗН должна:

1. Извлекать из текста условия предварительный набор спецификаций S_0 и предлагать его на проверку автору задачи. Результатом должен стать набор авторских спецификаций S.
2. Проверять заданные спецификации на непротиворечивость.
3. Извлекать из спецификации S граничные условия, особые точки, т.п.
4. Предлагать предварительное разбиение D_0 пространства данных на очевидные классы эквивалентности (например, базовым может стать соображение «длины переменной»).
5. Уточнять разбиение, основываясь на дополнительных сведениях, введенных автором задачи.

На втором этапе разработки ([2]) АПЗН должна:

1. Основываясь на заданном разбиении D, создавать наборы входных «половинок» тестов.
2. Если имеется авторский тестовый набор T_0, проверять его на полноту охвата заданного разбиения.

3. Проверять соответствие авторских тестов заданным спецификациям.

Если в процессе работы стала очевидной необходимость изменения или уточнения исходной спецификации S, то процесс создания тестового набора нужно начинать заново – уже с опорой на обновленную спецификацию S′.

Окончательный вариант тестового набора генерируется с использованием эталонного решения. На этом третьем этапе АПЗН должна:

1. Проверять соответствие полученных выходных данных спецификациям S.

При коллективной разработке задачного комплекта АПЗН должна предоставлять дополнительные возможности.

Предположим, что два человека $АР_1$ и $АР_2$ пишут два разных эталонных решения $ЭР_1$ и $ЭР_2$, используя для отладки два тестовых набора $Т_1$ и $Т_2$. Для каждого из них АПЗН производит описанные выше действия, после чего возникает ещё один этап сравнения и объединения этих тестовых наборов.

Оба решения $ЭР_1$ и $ЭР_2$ необходимо протестировать на объединенном тестовом наборе $Т = Т_1 \cup Т_2$. При отсутствии явных ошибок этот тестовый набор также должен пройти проверку на полноту охвата заданного разбиения и на соответствие спецификациям.

Вероятнее всего, объединенный тестовый набор будет избыточным. Следовательно, АПЗН должна определять, для какого из классов эквивалентности имеются «лишние» тесты, и предлагать авторам варианты сокращения набора.

Заметим, что при индивидуальной разработке задачного комплекта в роли наборов $Т_1$ и $Т_2$ могут выступать авторский тестовый набор и автоматически полученный тестовый набор.

Заключение

Система автоматизированной подготовки задачных наборов должна уметь оперировать разбиениями множества допустимых входных данных и тестовыми наборами как их отображениями.

В части разбиений:

1. Осуществлять текстовый анализ условия задачи с целью выделения явно заданных ограничений и спецификаций формата переменных.
2. Хранить разбиения в специально разработанном формате.
3. Предоставлять возможность задать или исправить разбиение вручную: добавлять, удалять, объединять или дробить классы эквивалентности.
4. Проверять разбиение на соответствие заданным спецификациям и ограничениям.
5. Сравнивать два разбиения.
6. Создавать объединения и пересечения разбиений.

В части тестовых наборов:

1. Генерировать входные данные на основе имеющегося разбиения.
2. Проверять, все ли классы эквивалентности из опорного разбиения представлены в некотором тестовом наборе.
3. Проверять набор на избыточность относительно опорного разбиения.
4. Объединять тестовые наборы (безусловное и условное объединения).
5. Пополнять и сокращать тестовые наборы.
6. Проверять соответствие тестов спецификациям.
7. Сохранять тестовые наборы.

Список литературы

1. Андреева, Т. А. Возможность автоматизации процесса генерирования тестовых наборов / Т. А. Андреева // Universum: Технические науки. – № 8(41). – Москва, 2017. – С. 5-7.

2. Андреева, Т. А. Генерирование тестовых наборов для автоматического тестирования / Т. А. Андреева // Материалы XXVII международной конференции «Современные информационные технологии в образовании». 28 июня 2016 г. Троицк – Москва, 2016. – С. 296-297.

Андреева, Т. А. Структура и классификация текстов олимпиадных

задач / Т. А. Андреева // Компьютерные инструменты в образовании. – № 3-4. – 2002. – С. 50-59.

Бейзер, Б. Тестирование чёрного ящика. Технологии функционального тестирования программного обеспечения и систем / Б. Бейзер. – Санкт-Петербург: Питер, 2004. – 320 с.

5. Кулямин, В. В. Обзор методов построения покрывающих наборов / В. В. Кулямин, А. А. Петухов. // Программирование, 37(3). – 2011. – С. 3-41.

6. Образовательный портал на базе интерактивной платформы для обучения детей, олимпиады по математике [Электронный ресурс] – Режим доступа: http://uchi.ru

7. ACM International Collegiate Programming Contest (Rules) / Available at: https://icpc.baylor.edu/worldfinals/rules

8. Manpreet Kapur, Rupinder Singh. A Review of Software Testing Techniques / International Journal of Electronic and Electrical Engineering. Volume 7, Number 5. – 2014. – P. 463-474.

УДК [002:004]:316.774

Андрюхина Т. В., Кордюкова Л. В., Третьяков А. Д., Третьякова Н. В.

ГОСУДАРСТВЕННАЯ ПОЛИТИКА В ОБЕСПЕЧЕНИИ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ ЛИЧНОСТИ

Татьяна Владимировна Андрюхина

Кандидат педагогических наук, доцент

sd-rgppu@mail.ru

ФГАОУ ВО «Российский государственный профессионально-педагогический университет», Россия, Екатеринбург

Лариса Владимировна Кордюкова

Кандидат искусствоведения, доцент