ОТЗЫВ

официального оппонента на диссертационную работу
Белеванцева Андрея Андреевича «Многоуровневый статический анализ исходного кода для обеспечения качества программ»,
представленную к защите на соискание ученой степени
доктора физико-математических наук по специальности 05.13.11 – математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей

Диссертационная работа Белеванцева А.А. посвящена проблеме обеспечения безопасности программного обеспечения и повышения его качества путем теоретического развития набора методов статического анализа поиска ошибок и уязвимостей в программах, а также практического применения этих методов в рамках единой методологии. Преимущества статического анализа как способа снижения количества ошибок в программе заключаются в поиске ошибок на всех путях исполнения программы, в том числе плохо покрываемых при тестировании, возможности применения к неполным программам при ежедневной разработке, разнообразию покрываемых классов ошибок. Но для разработки статического анализатора, реализующего эти преимущества при анализе современных программных систем в десятки миллионов строк кода, при этом укладываясь в 5-7 часов времени анализа и показывая высокий процент истинных срабатываний, нужно привлекать совокупность методов анализа, работающих совместно. Кроме того, требуется обеспечить это совместное функционирование в едином программном средстве и интерфейсе пользователя, позволяющем проводить регулярный анализ кода в жизненном цикле разработки программного обеспечения. Отсутствие удовлетворительного решения указанных задач и необходимость непрерывного совершенствования привлекаемых методов, возникающая из-за взрывного роста сложности анализируемых программ, определяет **актуальность** данной работы.

Диссертация состоит из введения, пяти глав и заключения. Объем диссертации составляет 229 страниц, включая 17 таблиц и 39 рисунков. Список литературы составляет 196 наименований.

Во введении обосновывается актуальность работы, формулируются цель и задачи работы, приводятся выносимые на защиту результаты.

Первая глава носит традиционный обзорный характер, концентрируясь на современных методах статического анализа, работающих с представлениями программы различного уровня и

проводящими как быстрый анализ путем поиска шаблонов на абстрактном синтаксическом дереве, так и глубокий межпроцедурный анализ с чувствительностью к контексту и путям выполнения. Также внимание уделяется перспективным направлениям развития анализаторов и особенностям применения анализаторов на практике.

Вторая глава посвящена предлагаемой методологии многоуровневого статического анализа исходного кода и содержит развитие математической теории анализа, которое заключается в формальном обосновании и доказательстве корректности центральной части методологии – совокупности методов анализа различных уровней. Глава начинается с описания методов анализа на абстрактном синтаксическом дереве (АСД). Для детекторов, выполняющих обходы АСД при поиске ошибок, предложена классификация, позволяющая сформулировать и доказать оценки их сложности.

Далее предлагается модель памяти, обеспечивающая полную совместимость с языками с адресной арифметикой, на основе которой выполняется как внутрипроцедурный анализ, так и дальнейшие более глубокие анализы. Алгоритмы построения модели памяти математически обосновываются, показывается их корректность и приводится оценка сложности.

Алгоритмы межпроцедурного контекстно-чувствительного анализа используют единую концепцию классов значений для отслеживания как целочисленных значений, так и указателей. Параметризация вычисленных данных по отношению к внешним ячейкам памяти позволяет создавать компактные аннотации функций для последующего межпроцедурного анализа с конфигурируемыми потерями точности (теорема 2.5, алгоритм 2.7).

Наконец, строятся алгоритмы чувствительного к путям анализа, расширяющие предыдущий уровень межпроцедурного анализа, а также непосредственно применяющие символьное выполнение с объединением состояний с помощью отслеживания необходимых предикатов, при которых вычисленные факты анализа имеют место. Предлагается способ упрощения предикатов по ходу анализа и доказывается корректность приведенных алгоритмов.

В третьей главе рассматривается часть предлагаемой методологии, заключающаяся в архитектуре программного средства Svace, в котором реализованы предложенные автором методы анализа. Разработанная архитектура решает задачу реализации в промышленном окружении всех описанных уровней анализа и задачу единообразной работы с набором анализаторов – их запуска, конфигурации и просмотра результатов. На первом этапе работы анализаторов прозрачно для пользователя строятся необходимые представления программы путем контролируемой сборки этой программы. В ходе этой сборки определяются необходимые события сборки (компиляции) и при их обработке запускаются собственные компиляторы,

разработанные на базе открытых компиляторов LLVM и OpenJDK, генерирующие нужные представления. На втором этапе организовывается параллельный межпроцедурный анализ программы. Наконец, на третьем этапе результаты работы записываются в контентно-адресуемое хранилище и отображаются в графическом интерфейсе с поддержкой разметки исходного кода и навигации по нему.

Четвертая глава посвящена детекторам для поиска ошибок всех уровней – детекторы уровня абстрактного синтаксического дерева, межпроцедурные детекторы без чувствительности к путям выполнения и чувствительные к путям детекторы. Для уровня АСД рассматриваются детекторы для Си/Си++ и детекторы, специфичные для Java и C#. На межпроцедурном уровне описываются детекторы основной части системы Svace, работающие для Си, Си++ и Java: приводятся события анализа, которые создаются реализующим алгоритмы второй главы ядром анализа для обработки детекторами, часто используемые детекторами атрибуты, детекторы критических ошибок разыменования нулевого указателя, использования памяти после освобождения, утечек памяти и ресурсов, проверкой помеченных данных и др. Среди чувствительных к путям детекторов описывается поиск переполнения буфера, деления на ноль, использования неинициализированных переменных. Отдельно описываются детекторы ошибок для программ на языке C#.

В пятой главе описываются экспериментальные результаты запусков программного средства Svace, включающие оценку затрат на выполнение контролируемой сборки, времени и памяти анализа, качества получаемых предупреждений (доле истинных срабатываний), размера генерируемых данных в хранилище. Оценка проводится на программах с открытым исходным кодом на всех поддерживаемых языках различных размеров от тысяч до миллионов строк исходного кода. Для исходного кода ОС Android 5 процент истинных срабатываний составляет примерно 70% для Си и Си++ и 80% для Java.

Заключение диссертации содержит основные результаты, выносимые на защиту, и наброски направлений дальнейших работ.

В диссертационной работе представлены следующие новые научные результаты:

- методология выполнения статического анализа исходного кода программ для поиска ошибок, заключающаяся в выполнении многоуровневого статического анализа с помощью разработанных набора моделей программы и методов межпроцедурного анализа с общей моделью памяти, поддерживающих чувствительность к путям и контексту выполнения. Разработанные методы применимы для языков программирования Си, Си++, Java, C#, теоретически обоснованы, математически

доказана их линейная масштабируемость и корректность;

- алгоритмы поиска конкретных ошибок в программе (детекторы) для разработанных методов анализа, которые выполняют поиск классов ошибок разного уровня критичности. На основе предложенной методологии и иерархии уровней анализа разработано более 200 детекторов, покрывающих большинство популярных типов ошибок. В частности, поддерживаются ошибки кодирования, неверного использования стандартных интерфейсов, разыменование нулевого указателя, переполнение буфера, ошибки управления памятью и ресурсами, использование неинициализированных переменных, ошибки многопоточных примитивов, недостижимый код и др.;
- архитектура программной системы, поддерживающая работу разработанных методов анализа в промышленном окружении и системах непрерывной интеграции, от построения внутреннего представления программы до показа результатов анализа, а также управление набором анализаторов для различных языков.

Практическая значимость работы подтверждается использованием разработанных программных средств Svace в коммерческих компаниях, включая компанию Самсунг и ее публичную систему разработки ОС «Тайзен».

Достоверность полученных результатов обусловлена корректностью математических определений и доказательств, апробацией результатов на международных и российских конференциях, работоспособностью разработанных программных продуктов. Автором опубликовано более 40 работ по методам анализа и оптимизации программ, получивших признание на международных симпозиумах и конференциях, использующих открытые компиляторы GCC и LLVM, которые легли в основу теоретических и практических разработок, выполненных в данной работе. Непосредственно по теме диссертации автором опубликовано 12 работ, из них – 10 в журналах списка ВАК, 4 – в журналах Web of Science. Получено 9 свидетельств о регистрации программ для ЭВМ. Некоторые вопросы, рассмотренные в диссертации, вошли в курс по анализу программ, читаемый автором на факультете ВМК МГУ им. Ломоносова.

В диссертационной работе следует отметить следующие недостатки:

1. Не все используемые термины теории компиляции введены в первой главе (например, чувствительность к путям выполнения).
2. Было бы интересно привести статистику АСД-детекторов, реализованных в Svace, по предложенной в главе 1 классификации.

3. Недостаточно детально описаны алгоритмы девиртуализации, применяемые для построения графа вызовов при анализе программ на Java и C# (разделы 3.3.5, 3.3.6 диссертации), а также их влияние на качество анализа.
4. Несомненно, 4 глава диссертации является существенной частью работы – детекторы для поиска ошибок, однако, на мой взгляд, ей следовало бы придать более систематичный характер. 200 детекторов – это много или мало? Вопрос полноты охвата возможных ошибок не рассмотрен. Хотелось бы видеть более систематическое изложение. При этом я признаю, что высокий уровень правильно найденных ошибок в реальных больших программах является, в какой-то мере, ответом на мое замечание.

Отмеченные недостатки не влияют на общую положительную оценку работы.

Диссертационная работа Белеванцева А.А. на соискание ученой степени доктора физико-математических наук является научно-квалификационной работой, в которой изложены научно обоснованные методы и созданы программные средства, внедрение которых вносит значительный вклад в развитие методов разработки программного обеспечения. Результаты диссертации представлены в статьях автора и докладывались на российских и международных конференциях. Автореферат правильно и полно отражает содержание диссертации.

Диссертационная работа отвечает требованиям ВАК РФ, предъявляемым к докторским диссертациям по специальности 05.13.11 – математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей, а ее автор, Белеванцев Андрей Андреевич, заслуживает присуждения ему ученой степени доктора физико-математических наук по специальности 05.13.11.

Официальный оппонент
заведующий кафедрой системного программирования
Федерального государственного бюджетного
образовательного учреждения высшего
образования «Санкт-Петербургский государственный университет»,
доктор физико-математических наук, профессор _____ А.Н. Терехов

30.01.2018