ОТЗЫВ
официального оппонента на диссертационную работу
Кошелева Владимира Константиновича

«Межпроцедурный статический анализ для поиска ошибок в исходном коде программ на языке C#»,

представленную к защите на соискание ученой степени кандидата физико-математических наук по специальности 05.13.11 – математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей

Проблема реализации надежных программ является чрезвычайно важной при разработке сложного программного обеспечения. Современные программные системы могут иметь большой размер (сотни миллионов строк кода) и могут применяться в критических прикладных областях, в которых программные сбои являются просто недопустимыми (как, например, в системах управления воздушным транспортом). В определенной мере данная проблема решается за счет самого широкого применения разработанных промышленных способов тестирования программ, однако данный подход является достаточно ресурсоемким, а исправление обнаруживаемых ошибок на стадии выполнения программ может потребовать значительных усилий. Кардинально иной подход состоит в попытке обнаружения допускаемых ошибок уже на стадии подготовки программного обеспечения при помощи статического анализа создаваемых программ. При этом принципиальными требованиями к методам и программным средствам статического анализа является максимально-возможный уровень обнаружения допускаемых ошибок, а также высокая производительность (анализ больших программ должен выполняться за время, приемлемое в практике разработки масштабного программного обеспечения). Тем самым, тема диссертационной работы Кошелева В.К. по разработке эффективных методов и программных средств статического анализа является **актуальной и значимой** как в отношении теории, так и практики программирования.

К **основным результатам**, полученным Кошелевым В.К. в ходе выполнения диссертационного исследования, следует отнести:

1. Алгоритм внутрипроцедурного статического анализа, учитывающий порядок и возможные пути выполнения операторов анализируемых программ.

2. Алгоритм межпроцедурного анализа, обеспечивающий возможность статического анализа для сложных многомодульных программ.

3. Критерий выдачи предупреждений, позволяющий управлять необходимым уровнем генерации истинных и ложных предупреждений.

4. Инструмент статического анализа, реализующий разработанные алгоритмы для анализа программ на языке C#, позволяющий проводить статический анализ больших программных систем, содержащих миллионы строк кода, за время, приемлемое при разработке сложного программного обеспечения.

Научная новизна результатов, представленных в диссертационной работе Кошелева В.К., заключается в разработке алгоритмов внутрипроцедурного и межпроцедурного статического анализа сложных многомодульных программных систем, в разработке критериев для управления необходимым уровнем генерации истинных и ложных предупреждений обнаружения дефектов анализируемого программного кода. Разработанные алгоритмы имеют строгое формальное обоснование, а разработанный программный инструмент проведения статического анализа и его практическая апробация с помощью выполненных вычислительных экспериментов показывает адекватность высокую эффективность разработанного подхода.

Общая структура диссертационная работа состоит из введения, шесть глав и заключения.

В **первой главе** проводится анализ научных работ, посвящённых проблеме поиска ошибок в исходном коде программ. Рассматриваются отличительные особенности методов статического анализа исходного кода, включая область их применимости. Делается вывод об использовании в качестве ос-

новного метода анализа комбинации символьного выполнения с межпроцедурным анализом, основанным на резюме. Обсуждаются возможности как коммерческих, так и открытых современных статических анализаторов.

Вторая глава посвящена описанию внутреннего представления, использующегося в качестве модели при анализе. Предложенное внутреннее представление учитывает особенности языка C#.

В **третьей главе** описывается метод символьного выполнения для проведения внутрипроцедурного анализа. Символьное выполнение осуществляет чувствительный к потоку и путям анализ исходного кода. Для ускорения анализа предлагается использовать объединение символьных состояний в точках слияния, а также оптимизировать размер предикатов пути и условий в точках слияния с помощью алгоритмов, использующих свойства доминирования. Для объединённого символьного состояния переопределяются правила интерпретации операции чтения и записи в поле. Для алгоритмов построения предикатов пути приводятся доказательства их корректности. Для циклов с фиксированным числом итераций рассматривается эвристика, позволяющая произвести выход из цикла после анализа первой итерации.

В **четвертой главе** рассматривается задача обощения результатов, полученных при внутрипроцедурном анализе, для проведения межпроцедурного анализа. Для каждого метода строится резюме, описывающее состояния памяти в терминах символьного выполнения в точке входа в метод и в точке выхода из метода. Такая модель позволяет соотнести состояние памяти вызывающего метода с состоянием памяти вызванного метода и, при необходимости, произвести обновления состояния памяти в точке после вызова. Кроме состояния памяти, в резюме предлагается сохранять также дополнительную информацию, необходимую детекторам ошибок.

В **пятой главе** обсуждаются различные варианты определения ошибочной ситуации и алгоритмы, способные их обнаружить. Приводится общее определение ошибочной ситуации и набор частных определений. Для каждого частного определения рассматривается пример исходного кода, иллюстри-

рующий шаблон обнаруживаемой им ошибочной ситуации. Рассматриваются алгоритмы поиска ошибок типа «доступ к нулевому указателю» и «утечка ресурсов». Обсуждаются причины нестрогости алгоритмов поиска ошибок.

В **шестой главе** рассматриваются аспекты реализации предложенных в работе методов на примере инструмента SharpChecker. Обсуждается построение по внутреннему представлению Roslyn таких структур, как граф вызовов и граф потоков управления. Обсуждается организация межпроцедурного анализа, включая стратегию обхода графа вызовов. Для оценки качества инструмента SharpChecker использовались проекты с открытым исходным кодом, размер которых достигал полутора миллионов строк кода. Максимальное время анализа среди рассматриваемых проектов составило 24 минуты. Также инструмент продемонстрировал хорошее качество обнаруживаемых дефектов.

Работа Кошелева В.К. выполнена на высоком научно-техническом уровне, однако имеются следующие замечания по диссертации:

1. В целом, диссертация хорошо изложена, однако есть ряд синтаксических ошибок (согласование окончаний слов в предложениях), достаточно часто используются аббревиатуры без их расшифровки (TIOBE, SAT, SMT, BMC и др.), встречаются жаргонные выражения (геттеры-сеттеры, синтаксический сахар,...), часть алгоритмов представлена не в алгоритмической форме (алгоритмы 3.1 и 3.2, алгоритм на стр. 55 не перенумерован).

2. В диссертационной работе рассмотрен поиск двух типов дефектов: «доступ к нулевому указателю» и «утечка ресурсов». Однако в работе не проводится анализ классов дефектов в программах на языке C#, которые могут быть найдены описанными методами.

3. Разработанный программный инструмент SharpChecker представлен в очень кратком виде, практически нет описания данного инструмента как программного продукта.

4. В приведенных результатах апробации разработанного подхода (глава 6) отсутствует сравнение программного инструмента SharpChecker с какими-либо уже существующими системами статического анализа программ.

Данные замечания не снижают обоснованности выводов данной работы и не влияют на положительную оценку полученных результатов. Как заключение можно утверждать, что в диссертационной работе В.К. Кошелева выполнены все требования ВАК, предъявляемые к диссертациям на соискание ученой степени кандидата физико-математических наук в соответствии с п. 9 положения ВАК о присуждении ученых степеней, утвержденного постановлением Правительства РФ от 24.09. 2013 № 842 (ред. от 30.07.2014). Содержание диссертационной работы полно и правильно отражено в автореферате. Диссертация представляет собой законченную научно-исследовательскую работу, а Кошелев Владимир Константинович заслуживает присуждения ему ученой степени кандидата физико-математических наук по специальности 05.13.11 – математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей.

Директор Института информационных технологий,
математики и механики Национального исследовательского
Нижегородского государственного университета имени Н. И. Лобачевского,
зав. кафедрой Программной инженерии, д.т.н., проф.
г. Нижний Новгород, пр. Гагарина, 23
Тел.: (831) 462-30-85, gergel@unn.ru

Гергель В.П.