ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«НОВОСИБИРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ» (НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ, НГУ)
Факультет информационных технологий
Кафедра параллельных вычислений
09.03.01 Информатика и вычислительная техника.
Программная инженерия и компьютерные науки

СТАТИЧЕСКИЙ АНАЛИЗ ФРАГМЕНТИРОВАННЫХ ПРОГРАММ НА БАЗЕ ГРАФА ЗАВИСИМОСТЕЙ ПО ДАННЫМ

Выполнил: Царев Василий Дмитриевич, студент группы 20202 ФИТ НГУ
Руководитель ВКР: Власенко Андрей Юрьевич, к.т.н., доц. каф. ПВ ФИТ НГУ
Соруководитель ВКР: Перепёлкин Владислав Александрович, к.т.н., ст. преп. каф. ПВ ФИТ

Новосибирск 2024 24.06.2024

Система LuNA

Система LuNA (Language for Numerical Algorithms) автоматизирует создание параллельных программ по описанию фрагментированных алгоритмов.

Фрагмент данных (ФД) – единица информации. Может быть индексированным.

Фрагмент вычислений (ФВ) – порождаемая во время выполнения программы единица работы.

Фрагмент кода (ФК) – пользовательская процедура. Бывают атомарные и структурированные.

Особенности:

- Недетерминированный порядок выполнения ФВ.
- ФД могут быть инициализированы лишь единожды.

Актуальность

Написание программ на языке LuNA сопряжено с риском допущения специфических семантических и синтаксических ошибок.

- Поиск семантических ошибок трудозатратен, и на текущий момент готовых к использованию средств отладки для языка LuNA не существует.
- Синтаксические ошибки не будут пропущены компилятором, однако в настоящее время компилятор системы LuNA не всегда предоставляет достаточную и легко воспринимаемую информацию об ошибках, которая необходима для их устранения.

Предметная область

Статический анализ – анализ исходного кода программы без её запуска, в противоположность динамическому анализу.

Применяется во время всего цикла разработки программы и уменьшает трудозатраты на поиск ошибок. Не требует перекомпиляции и не искажает поведение программы.

Зачастую используется **промежуточное представление** исходной программы.

Существует множество различных промежуточных представлений.

Цель работы и задачи

Цель работы:

Разработать алгоритмы статического анализа LuNA-программ для наиболее типовых классов семантических ошибок, опираясь на графовые представления программ, и реализовать данные алгоритмы в программном средстве.

Задачи:

- Принять участие в формировании базы типов ошибок в LuNA-программах.
- Провести обзор методов и средств статического анализа.
- Выбрать (и обосновать выбор) промежуточного представления LuNA-программ для проведения статического анализа с целью выявления семантических ошибок.
- Разработать и протестировать модуль статического анализа с использованием выбранного представления.
- Интегрировать разработанный модуль в комплекс автоматизированной отладки LuNA-программ ADAPT.

Обзор средств и методов статического анализа

Сторонние анализаторы:

- cppcheck (C, C++)
- PVS Studio (C, C++, C#, Java)
- SonarLint (Java)
- Stan (Haskell)
- weeder (Haskell)

Существующие средства **не могут быть применены для решения задачи**, поскольку не учитывают особенности системы LuNA, не предоставляют возможность обнаруживать семантические ошибки или не могут работать с синтаксисом языка.

Обзор средств и методов статического анализа

Промежуточные представления:

- *Абстрактное синтаксическое дерево* (AST, abstract syntax tree). Является трудноприменимым для обнаружения многих типов семантических ошибок.
- *Граф потока управления* (CFG, control-flow graph). Использовать его не представляется возможным вследствие недетерминированности выполнения ФВ.
- *Граф зависимостей по данным* (DDG, data dependence graph). Является перспективным, поскольку он может быть однозначно построен исходя из требований ФВ по входным ФД и предоставляет возможность обнаруживать различные семантические ошибки.
- …

Требования к разрабатываемому программному средству

- Наличие функций для автоматического построения графа зависимостей по данным произвольной LuNA-программы.
- Наличие функций, использующих граф зависимостей по данным для поиска ошибок в программе.
- Наличие функций, позволяющих генерировать отчёты об ошибках в унифицированном формате.
- Наличие консольного интерфейса.

Предлагаемое решение

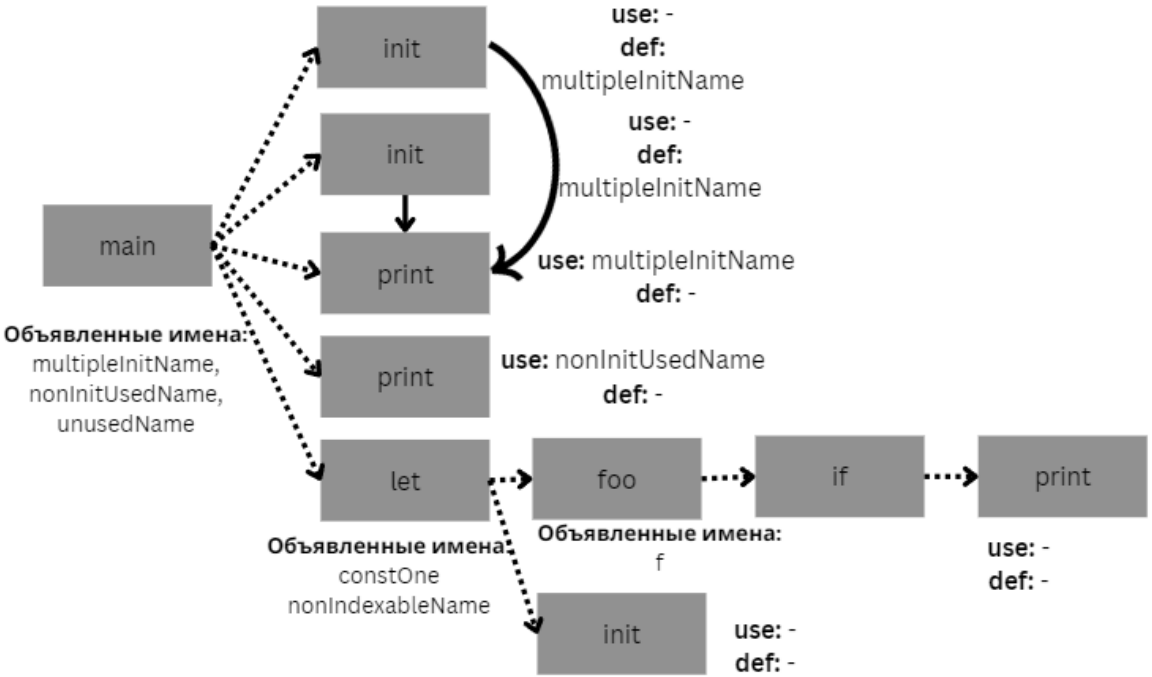
- Адаптировать граф зависимостей по данным под программы на языке LuNA. Вершинами графа станут операторы, а рёбра будут указывать зависимости по ФД между ними.
- Автоматизировать построение графа из AST входной программы, которое было создано в другой работе и позволяет анализировать даже синтаксически некорректные программы.
- Использовать обход графа и сбор релевантной информации для вывода пользователю отчёта об ошибках.

Разработанное промежуточное представление

Вершины – операторы языка LuNA.

Пунктирные рёбра (типа *“inside”*): нахождение операторов в блоках друг друга.

Сплошные рёбра (типа *“use-def”*): зависимости по ФД между операторами.



Разработка алгоритмов поиска ошибок

Поскольку ошибки различных типов требуют различных методов поиска, было принято решение разработать **базу типов ошибок** совместно с другими студентами. На текущий момент база содержит 37 типов ошибок.

Разработанные алгоритмы поиска ошибок включают в себя:

- линейный обход списка всех вершин и проверки на наличие всех требуемых входящих и исходящих дуг (зависимостей по данным);
- анализ значений идентификаторов и свёртка констант;
- обход графа в глубину по дугам типа “inside” и “use-def”.

Реализация

Был создан модуль, называемый **DeGSA** (Dependence Graph Static Analyzer). Исходный код написан на языке C++. Он способен обнаруживать ошибки девяти различных типов.

Работа модуля состоит из 4 этапов.

- Анализ сигнатур ФК.
- Порождение вершин.
- Связывание вершин.
- Поиск ошибок.

Синтаксические ошибки обнаруживаются на **2** этапе, *семантические* – на **4**.

Реализация

Список обнаруживаемых DeGSA ошибок:

Синтаксические:

- LUNA02. Несуществующая LuNA-подпрограмма.
- LUNA13. ФД с одинаковыми названиями в одной области видимости.
- LUNA14. Попытка использования необъявленного ФД.
- LUNA26. Попытка инициализации объекта, не являющегося ФД.
- LUNA36. Попытка индексации объекта, не являющегося ФД.

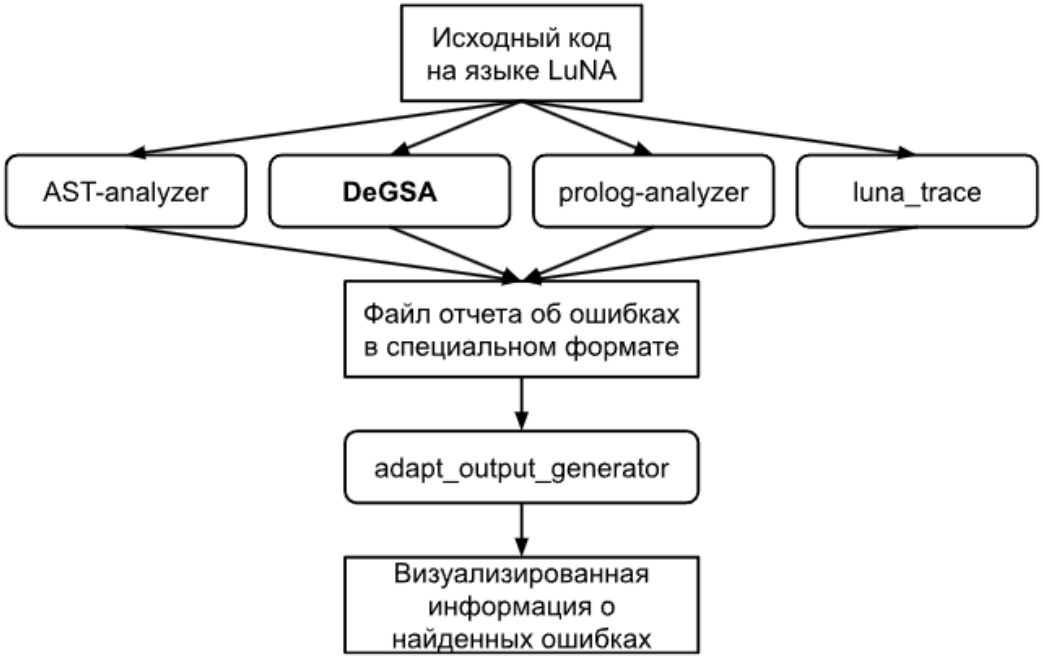
Семантические:

- LUNA03. Повторная инициализация ФД.
- LUNA05. Попытка использования неинициализированного ФД.
- LUNA10. Неиспользуемое имя.
- LUNA23. Формула в if тождественно истинна/ложна.

Интеграция в комплекс ADAPT

Комплекс автоматизированной отладки **ADAPT** (Analyzer, **Dependence graph**, AST, Prolog, Trace).

- Автоматизирована генерация файлов JSON разработанного унифицированного формата с информацией об ошибках.
- Внесён вклад в разработку утилиты по генерации текстовых отчётов на основе JSON-файлов.



Тестирование

Модуль DeGSA был протестирован на корректность и на скорость работы.

1. **Корректность** – для каждого из девяти обнаруживаемых модулем типов ошибок была создана соответствующая программа. Модуль вернул корректную информацию о каждой ошибке в каждой из девяти программ.
2. **Скорость** – была использована программа, моделирующая поведение протопланетного диска и содержащая более 10 000 строк. Время работы модуля составляло от 54.7 до 57.1 секунды, что является приемлемым показателем (проведено 5 экспериментов).

Заключение

- Был разработан и реализован модуль статического анализа, применяющий граф зависимостей по данным и обнаруживающий ряд синтаксических и семантических ошибок с последующим выводом информации пользователю.
- Созданный модуль был интегрирован в комплекс автоматизированной отладки ADAPT и протестирован.
- Принято участие в заполнении базы типов ошибок в программах на языке LuNA.
- Внесён вклад в разработку шаблонов описания ошибок и генерацию отчётов комплексом ADAPT.

Апробация и публикации

- Всероссийская летняя XLI молодежная Школа-конференция по параллельному программированию (3 - 14 июля 2023 г.). Царев В. Д., “Автоматическое создание графа зависимостей для статического анализатора LuNA-программ” (тезисы и доклад, принято в печать, получена справка).
- Молодежные Марчуковские научные чтения 2023 (18 - 20 декабря 2023 г.). Царев В. Д., “Статический анализ фрагментированных программ на базе графа зависимостей по данным” (тезисы и доклад, принято в печать, получена справка).
- 62-ая Международная научная студенческая конференция 2024 (17 - 23 апреля 2024 г.). Царев В. Д., “Статический анализ фрагментированных программ на базе графа зависимостей по данным” (тезисы и доклад, принято в печать, получена справка).
- Вестник НГУ. Серия “Информационные технологии”. Том 22, выпуск 1. Власенко А. Ю., Мичуров М. А., Царев В. Д., Курбатов М. А., “Построение комплекса автоматизированной отладки фрагментированных программ” (статья, принято в печать, получена справка).
- Выступление на семинаре кафедры ПВ ФИТ (доклад).

Научная новизна и практическая значимость

Научная новизна работы заключается в том, что был предложен метод статического анализа фрагментированных программ на базе графа зависимостей по данным с целью обнаружения семантических ошибок.

Практическая значимость работы заключается в возможности сократить трудозатраты программистов на языке LuNA с помощью автоматизации процесса отладки программ. Выводимая анализатором информация легко воспринимается и позволяет быстро устранить ошибку.

Спасибо за внимание!

Дальнейшие планы

Планируется:

- Расширить список типов обнаруживаемых модулем DeGSA ошибок.
- Улучшить алгоритмы поиска ошибок, связанных с индексированными ФД, с помощью анализа значений индексов.
- Разработать способы сокращения времязатрат на проведение анализа.
- Внедрить анализ и защиту от рекурсии в исходном коде входной программы.

Заключение

Защищаемые положения:

- Разработано промежуточное представление для LuNA-программ.
- Созданы алгоритмы поиска семантических ошибок с помощью разработанного промежуточного представления.
- Реализовано автоматическое построение и применение промежуточного представления.
- Проведены интеграция в комплекс автоматизированной отладки ADAPT и тестирование созданного модуля.
- Внесён вклад в разработку базы типов ошибок в LuNA-программах и в реализацию комплекса автоматизированной отладки ADAPT.