**Министерство науки и высшего образования Российской Федерации**

**федеральное государственное автономное образовательное учреждение высшего образования**

**“НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО”**

**Факультет** Программной инженерии и компьютерной техники

**Образовательная программа** Системное и прикладное программное обеспечение

**Направление подготовки (специальность)** 09.04.04 Программная инженерия\_\_\_\_\_\_\_\_\_

**Отчет**

**о научно-исследовательской работе**

Тема задания: «Выработка методов к анализу мультиязыковых текстов программ»

Обучающийся: Орловский М.Ю P4116

(Фамилия И.О.) (номер группы)

Руководитель практики от университета: Маркина Т.А, доцент факультета ПИиКТ

Санкт-Петербург

2023 г.

# Эпоха

## Исследовать возможность обобщения предлагаемого метода на сочетание различных языков программирования.

В ходе подготовки статьи на конференцию «КМУ» была проведена небольшая ревизия метода анализа, для его большей формализации. Было выяснено, что структуру семантического узла можно обобщить, если немного изменить свойства его компонентов. В данный момент, структура узла представляет информацию, представленную в таблице 1.

|  |  |  |  |
| --- | --- | --- | --- |
| Таблица 1 – Обновленная структура семантического узла | | |  |
| Имя поля | Тип поля | Семантика | Что обновлено |
| Позиция | Индекс AST узла | Указывает позицию анализируемой синтаксической конструкции | Было решено, что будет разумнее рассматривать позицию как конкретный AST узел из исходного дерева кода |
| Атрибуты | Гетерогенный список | Содержит список атрибутов узла | В отличие от предыдущего варианта хранения атрибутов (строковый ключ – строковое значение), гетерогенный список является более гибким механизмом, который проще обрабатывать программно |
| Вид | Перечисление | Является способом типизации конкретного семантического узла, исходя из ЯП, который он описывает | В текущей версии не требует изменений |
| Семантика | Перечисление | Используется для отражения семантики, представляемой узлом | В текущей версии не требует изменений |

Таким образом, предполагается, что данная структура узла будет более гибкой в отношении способов хранения информации о семантике конкретных синтаксических конструкций.

Что касается алгоритма связывания, то он не претерпел явных изменений. В текущем виде поля «вид» и «семантика» семантического узла используются для группировки узлов при связывании. Алгоритм при этом имеет сложность O(n2)

## Исследовать наиболее часто используемые парадигмы программирования (процедурное, ОО, декларативное) в контексте мультиязыкового анализа

Прежде чем рассматривать мультиязыковой анализ в контексте обрабатываемых в нем языков, было бы полезно обобщить и категорировать эти языки. В целях работы предлагается использовать общепринятое разделение языков по парадигмам [1]. Данное разделение представлено в таблице 2.

|  |  |  |
| --- | --- | --- |
| Таблица 2 – Разделение языков по парадигмам | | |
| Парадигма | Основные особенности | Представители |
| Императивная | Переменные как способ манипуляции памятью; Мутабельность как основной способ проведения вычислений; Описание вычислений как конкретного набора инструкций | C, Pascal, Java, Golang, Bash, JavaScript и многие другие |
| Декларативная | Описание результата и его свойств, как способ его получения; Описание вычислений как высокоуровневых объявлений; | ML, Lisp, Haskell, SQL, Make, HTML и также многие другие |

Несмотря на то, что такого разделения достаточно для категоризации языков, оно будет расширено и детализировано, так как в рамках данной работы предстоит рассматривать различные языки очень разной природы. Такое расширенное и уточненное разделение представлено в таблице 3.

|  |  |  |
| --- | --- | --- |
| Таблица 3 – Расширенное разделение языков по парадигмам | | |
| Парадигма | Основные особенности | Представители |
| Процедурная и структурная | Инструкции сгруппированы в процедуры/подпрограммы; переменные как способ взаимодействия с памятью | C, Pascal, Golang, Bash, и многие другие |
| Объектно-ориентированная | Данные и код инкапсулированы в единое представление – объект; Объекты общаются посредством передачи сообщений друг-другу | Java (отчасти), C# (отчасти), JavaScript (отчасти), C++ (отчасти), Smalltalk, Erlang, Elixir, Self и другие |
| Функциональная | Функции, как способ выстроить процесс вычисления; Иммутабельность как фундамент формальных гарантий в коде | ML, Rust (отчасти), Haskell, Erlang, Elixir, Lisp |
| Макро | Процесс вычислений проводится путем лексических подстановок строк для достижения конечной строки; встречаются варианты, оперирующие на AST | Препроцессор C, Make, макросистема Rust, различные DSL в Web фреймворках |

Исходя из данной классификации, можно разделить анализируемые языки по категориям, в каждой из которых можно будет сформировать свою онтологию, описывающую основные особенности языков данной категории и их основные компоненты. Стоит учесть, что данное разделение сделано в целях упрощения и обобщения анализа и может не соответствовать общепринятому.

## Рассмотреть различные подходы к представлению семантической информации

Основным способом представления информации (и самым универсальным) является семантическая сеть. В данной работе был избран именно такой подход, однако, семантические сети трудно структурировать и формализовывать. И хотя в дальнейшем будет использоваться именно этот подход к представлению семантической информации, полезно рассмотреть иные подходы, активно используемые в компиляторах, оптимизаторах и анализаторах. Такие подходы отражены в таблице 4.

|  |  |  |  |
| --- | --- | --- | --- |
| Таблица 4 – Представление информации об исходном коде | | |  |
| Название представления | Основные особенности | Плюсы (в рамках данной работы) | Минусы (в рамках данной работы) |
| AST | Представление иерархической структуры программы, посредством связывания её синтаксических конструкций в единое дерево | Легко анализируется, может быть использовано в анализах, где порядок анализа не важен; легко визуализируется; универсально для многих видов анализа | Сложно изменяется; не подходит для анализов, которые зависят от порядка; некоторые анализы неприменимы |
| CFG | Представление пути исполнения программы, моделируется связями (переходами) между узлами (операторами или выражениями) | Многие другие анализы легко реализуются через CFG; хорошо подходит для императивных языков | Плохо подходит для неимперативных языков, где путь исполнения программы задан неявно; сложнее реализовать чем AST |
| Семантическая сеть | Представление сущностей и их зависимостей в виде графа, узлы которого определяют сущности, а связи - отношения | Легко синтезируется; очень универсально; очень гибко и может быть использовано в различных анализах | Слабо структурировано; нет четкой схемы как связывать узлы между собой, по какому принципу выявлять сущности и отношения |
| Иерархия типов | Представление информации о связях сущностей в виде типов этих сущностей, формирующих определенную иерархию | Может быть простым вариантом представления большого объема информации в ОО и функциональных системах; имеет много формальных средств и методов для анализа | Сильно зависит от избранного языка; может не нести ценности, если язык имеет слабую типизацию; довольно сложна для анализа в общем случае |
| Онтология | Представление, основанное на формальных знаниях в конкретной предметной области | Однажды специфицированная онтология может быть использована в другом языке при наличии схожих семантик; универсальна и расширяема | Быстро становится сложной при расширении; обычно, предметные области (а в особенности ЯП) трудно формализуемы |

Стоит заметить, что существует еще большое множество иных представлений, которые слабо подходят для описания семантики языков программирования: фреймовые, логические и вероятностные модели, а также нейронные сети. Хоть они и представляют ценность для реализации различных анализов, в рамках данной работы они либо слишком сложны для применения, либо не обеспечивают необходимой точности.

Исходя из перечисленных в таблице 4 подходов к представлению семантической информации, можно заключить, что использование семантических сетей стоит совместить с иными способами представления информации. Предполагается, что использование онтологий упростит формирование семантических сетей, позволит их формализовать и повысит количество переиспользуемых техник анализа.

# Эпоха

###### ЗАКЛЮЧЕНИЕ

В ходе выполнения научно-исследовательской работы были проведены анализ и оценка методов к осуществлению мультиязыкового анализа, выявлены их достоинства и недостатки, а также проведен анализ существующих анализаторов мультиязыковых проектов.

Были выявлены ключевые характеристики мультиязыкового анализа и выстроена универсальная семантическая структура для представления данных анализатора. Выполнена разработка прототипа мультиязыкового анализатора, ориентированного на использование в реальном времени с учетом последующей интеграции в IDE. Проведено его тестирование и оценка результатов анализа.

###### СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

https://homes.cs.aau.dk/~normark/prog3-03/html/notes/paradigms\_themes-paradigm-overview-section.html