Статический вывод типов для языка Python в интегрированной среде разработки

Автор: Мирошников Владислав Игоревич, 19.Б11-мм
Научный руководитель: доцент кафедры СП, к.ф.-м.н. Булычев Д. Ю.
Консультант: инженер ключевых проектов ООО «МПГ Айти Солюшнз» Захаров А.А.
Рецензент: ведущий инженер ключевых проектов ООО «Техкомпания Хуавей» Лукьянова О.Е.

Санкт-Петербургский Государственный Университет
Кафедра системного программирования

Введение

- Неотъемлемой частью любой среды разработки является кодовая модель (Code Model)
- SRC IDE — IDE, поддерживающая Java и разрабатывающаяся в данный момент для Python
- Вывод типов в IDE — одна из ключевых и важных задач в подсистеме кодовой модели
- **Мотивация:** вывод типов необходим для различной функциональности, например, подсистемы автодополнения или анализа семантики программы
- Предлагается реализовать статический вывод типов для языка Python и внедрить в разрабатываемую Python IDE

Цель и задачи

Цель: реализация статического вывода типов для языка Python в рамках добавления поддержки Python в SRC IDE

Задачи:

1. Изучить систему типов языка Python, а также существующие целевые решения, позволяющие статически выводить типы
2. Рассмотреть различные подходы и алгоритмы вывода типов для языка Python, а также выбрать наиболее подходящий для дальнейшей реализации с учетом выявленных требований и ограничений
3. На основе выбранного подхода спроектировать систему типов в рамках Python IDE, учитывающую особенности и свойства типовой системы Python
4. Реализовать подсистему статического вывода типов, принимая во внимание особенности архитектуры Python IDE
5. Разработать инфраструктуру тестирования и провести сравнения с аналогами

Система типов языка Python

- Система типов времени исполнения
  - Все является объектами
  - Объекты имеют тип, а не переменные
  - Стандартная иерархия типов
  - «Утиная» типизация (duck typing)

- Система типов постепенной типизации (gradual typing)
  - Богатый набор аннотационных конструкций
  - Не оказывает влияния на процесс работы программы
  - Единица типизации — переменная, а не объект
  - Номинативная и структурная типизации

Существующие решения

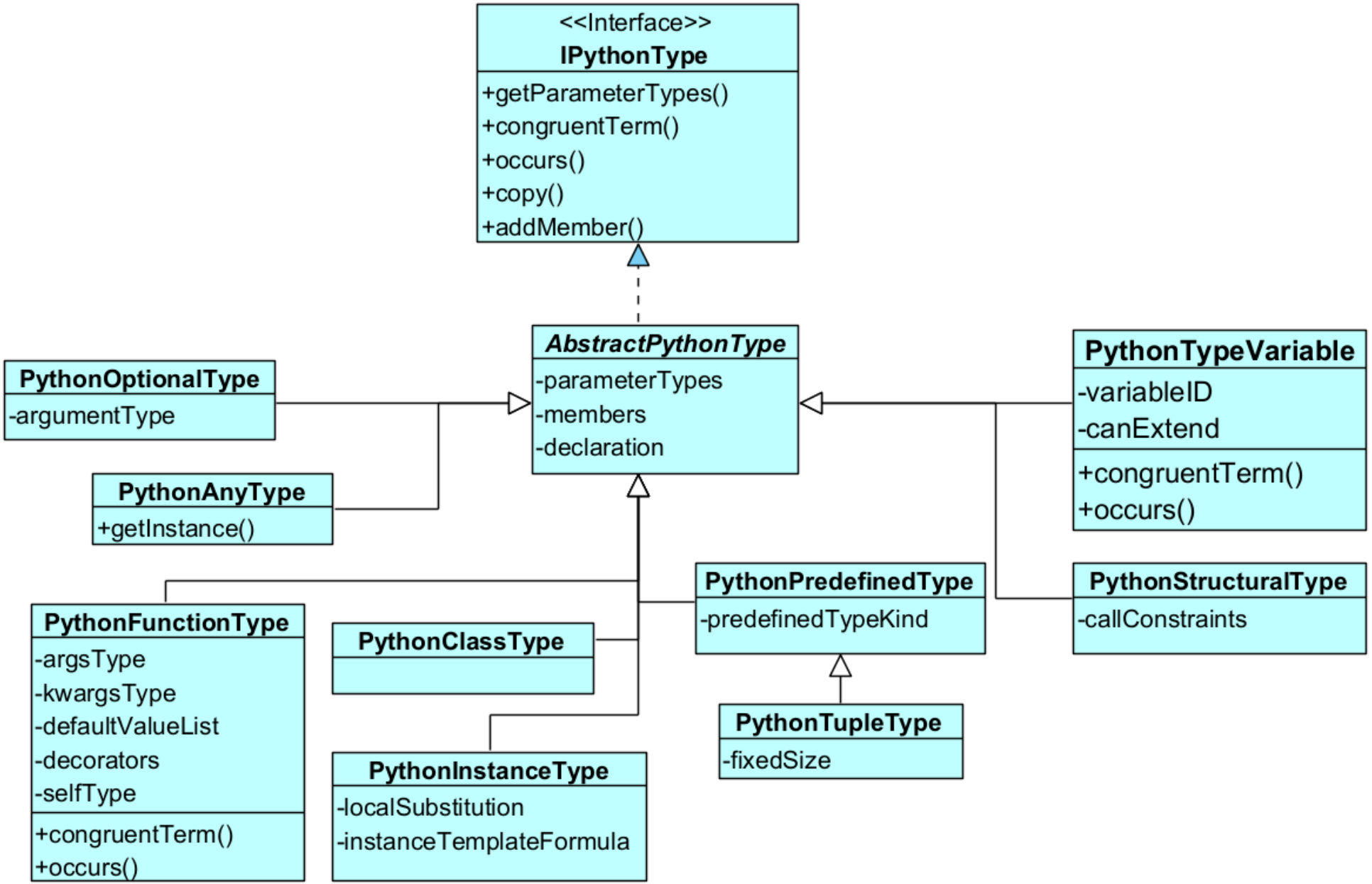
- Решения, интегрированные в IDE или компоненты кодовой модели
  - PyCharm
  - Jedi
  - Pyright
- Статические анализаторы Python кода
  - Mypy
  - Pytype

Алгоритмы вывода типов

- Алгоритмы машинного обучения
- Вероятностный вывод типов
- Алгоритм декартового произведения
- Вывод типов с системой верхних/нижних ограничений
- Вывод типов с использованием унификации

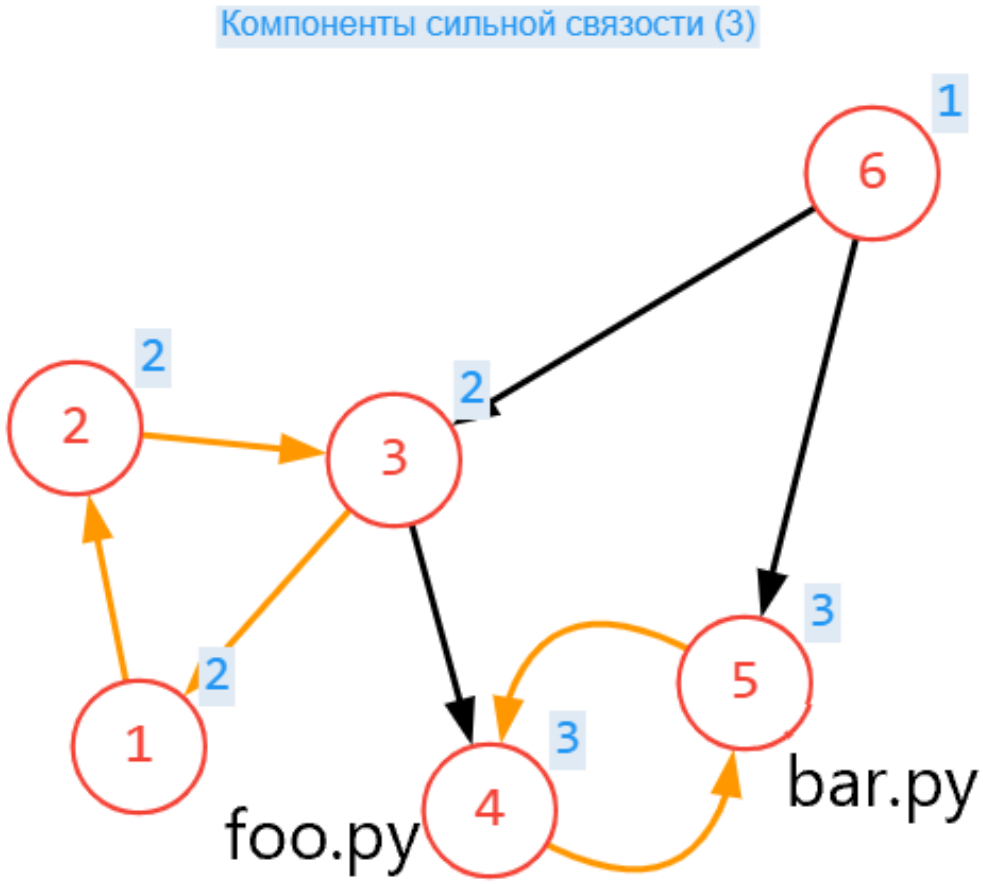
Ключевое решение: вывод типов с использованием унификации первого порядка и системой с роу-полиморфизмом

Архитектура системы типов



Принцип работы подсистемы вывода типов

- Индекс вывода типов инициирует создание групп вывода
- Группы вывода строятся на основе графа зависимостей модулей – граф компонент сильной связности
- Построение графа на основе индекса импортов – устойчивость к циклическим импортам
- Инкрементальное обновление типов в рамках изменившейся компоненты



Вывод в рамках группы вывода

- Начальный этап – создание формул для деклараций
- Составление списка задач и добавление «действий вывода»
- Итеративный процесс вывода типов с использованием алгоритма, вычисление и применение множества подстановок, сбор и обработка динамических «ограничений» (dynamic constraints)
- Полиморфный вывод типов при вызовах функций и импортах
- Условие остановки – неподвижная точка
- Замыкание формул – вычисление итоговой параметризации классов и экземпляров класса

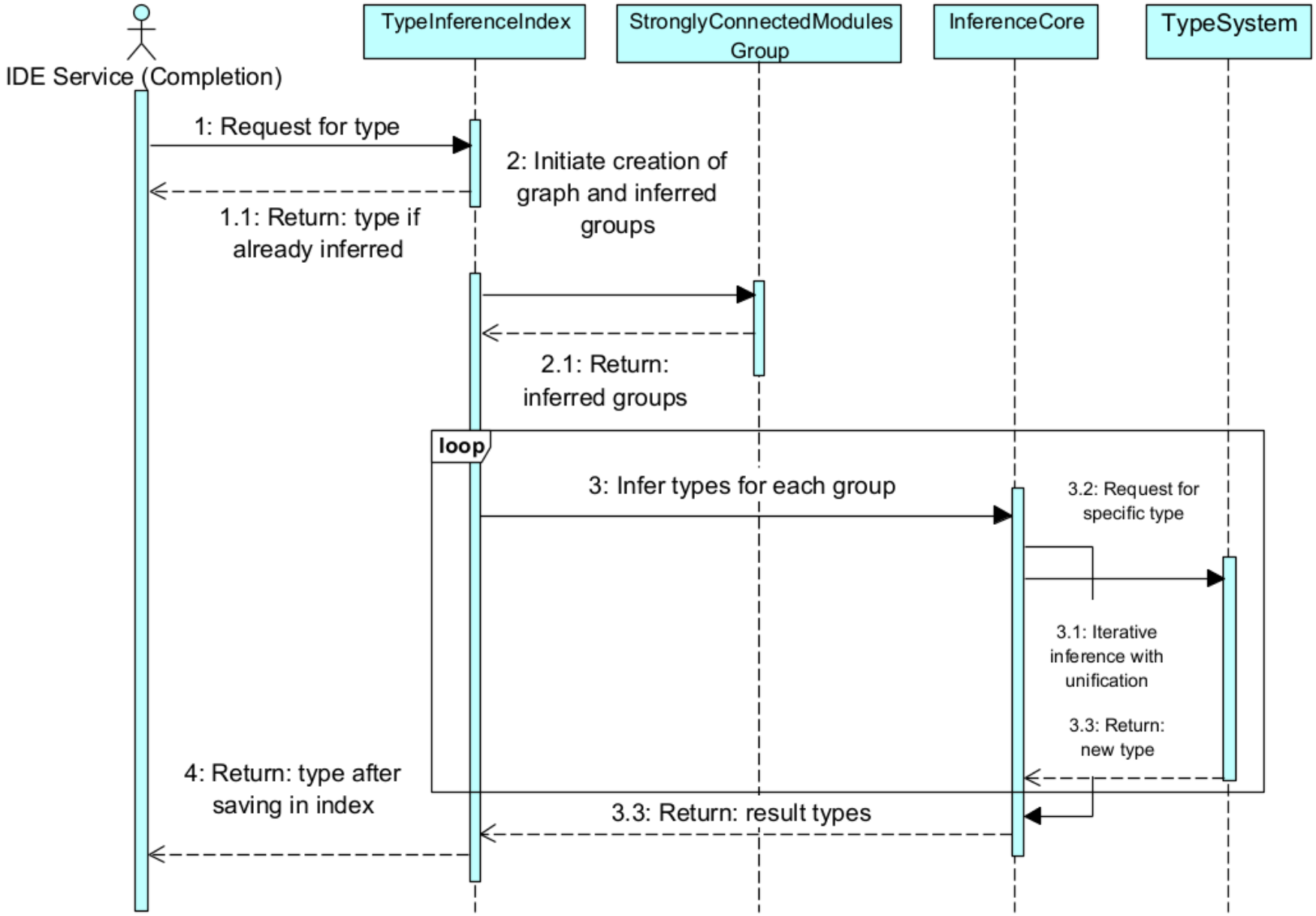
Алгоритм унификации с использованием роу-полиморфизма

- Базис – классический алгоритм синтаксической унификации Хиндли-Милнера
- Четыре фундаментальных операции:
  - walk
  - occurs check
  - extend substitution
  - congruent term
- Унификация на основе определенных правил:

$$\alpha \ \& \ A[int, Dict[str, \beta]] = \alpha \rightarrow A[int, Dict[str, \beta]]$$

  - Унификация функций – унификация их параметров
  - Ошибка унификации – переход типа в Any
- Роу-полиморфизм:
  - Унификация атрибутов
  - Расширение структурного типа

Сценарий использования подсистемы

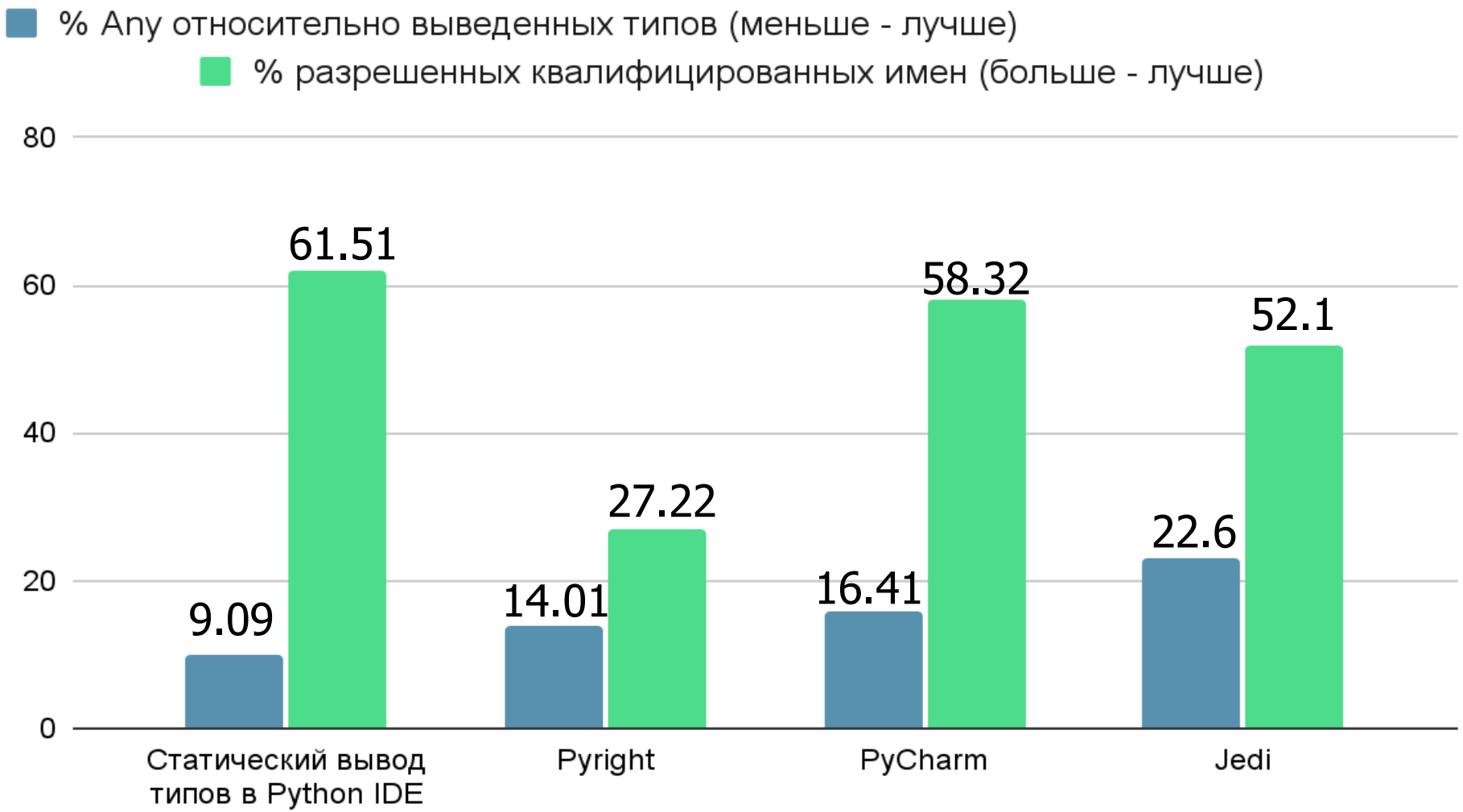


Апробация и тестирование решения

- Создана тестовая инфраструктура для преобразования типов
- Реализована конвертация типов, добавлена функциональность для проведения регрессионного тестирования
- Качественные сравнения поддержки «утиной» типизации

Апробация и тестирование решения

- Количественные сравнения с существующими решениями на библиотеке Numpy



Результаты

1. Изучена система типов языка Python, рассмотрены существующие целевые решения, выявлены ключевые недостатки данных решений
2. Рассмотрены подходы и алгоритмы вывода типов, выбран подход на основе теории унификации и системы с роу-полиморфизмом
3. Спроектирована система типов языка, предоставляющая структурные типы, параметрические и функциональные типы
4. Реализована подсистема статического вывода типов языка Python в Python IDE, учитывающая вывод для типов с атрибутами, а также параметризацию классов и функций
5. Разработана инфраструктура тестирования, добавлена возможность регрессионного тестирования, выполнены сравнения с существующими решениями, показывающие эффективность данного подхода