УДК 004.4 https://doi.org/10.26160/2541-8637-2026-17-207-211

ПОДХОДЫ К СИНТАКСИЧЕСКОМУ И ЧАСТИЧНОМУ СЕМАНТИЧЕСКОМУ АНАЛИЗУ ЯЗЫКОВ ПРОГРАММИРОВАНИЯ С ДИНАМИЧЕСКОЙ ТИПИЗАЦИЕЙ

Саитов Р.И.
Омский государственный технический университет, Омск

Ключевые слова: динамическая типизация, статический анализ, синтаксический анализ, семантический анализ, абстрактное синтаксическое дерево, частичный семантический анализ, контекстно-зависимые методы, эвристический анализ.
Аннотация. Актуальность работы обусловлена сложностью статического анализа программ на языках с динамической типизацией, где значительная часть семантической информации раскрывается только на этапе выполнения. Традиционные методы, ориентированные на строгую типизацию, недостаточно эффективны из-за отсутствия явных типов и динамического связывания. Цель исследования – анализ подходов к синтаксическому и частичному семантическому анализу динамически типизированных языков, а также обоснование роли абстрактного синтаксического дерева как основы для последующего анализа. Рассмотрены этапы лексического и синтаксического разбора, построение абстрактного синтаксического дерева и методы извлечения семантической информации в условиях неполной типовой определённости. Показано, что применение эвристических и контекстно-зависимых методов (анализ областей видимости и потоков данных) повышает точность выявления ошибок (необъявленные идентификаторы, нарушения областей видимости). Полученные результаты систематизируют подходы к статическому анализу и могут быть использованы при разработке средств поддержки программирования и автоматической обработки кода.

APPROACHES TO SYNTACTIC AND PARTIAL SEMANTIC ANALYSIS OF DYNAMICALLY TYPED PROGRAMMING LANGUAGES

Saitov R.I.
Omsk State Technical University, Omsk

Keywords: dynamic typing, static analysis, syntactic analysis, semantic analysis, abstract syntax tree, partial semantic analysis, context-dependent methods, heuristic analysis.
Abstract. The relevance of this work stems from the complexity of static analysis of programs in dynamically typed languages, where a significant part of semantic information is revealed only at runtime. Traditional methods designed for strict typing are insufficiently effective due to the absence of explicit types and dynamic binding. The aim of the study is to analyze approaches to syntactic and partial semantic analysis of dynamically typed languages, as well as to substantiate the role of the abstract syntax tree as a basis for subsequent analysis. The stages of lexical and syntactic parsing, abstract syntax tree construction, and methods for extracting semantic information under conditions of incomplete type certainty are considered. It is shown that the use of heuristic and context-dependent methods (scope and data flow analysis) improves the accuracy of detecting errors (undeclared identifiers, scope violations). The obtained results systematize approaches to static analysis and can be used in the development of programming support tools and automatic code processing.

I. Введение

Языки с динамической типизацией широко применяются благодаря гибкости, однако отсутствие статически определённых типов и динамическое связывание усложняют статический анализ кода. Традиционные методы,

ориентированные на строгую типизацию, в таких условиях ограниченно применимы, поскольку синтаксический анализ даёт лишь структурное представление, не учитывая контекст использования идентификаторов [1]. Возникает необходимость в методах частичного семантического анализа, основанных на структуре и контексте программы. Актуальна задача исследования таких подходов для получения семантически значимой информации на этапе статической обработки.

II. Постановка задачи

Цель исследования – анализ подходов к синтаксическому и частичному семантическому анализу языков программирования с динамической типизацией, а также обоснование использования абстрактного синтаксического дерева в качестве базовой структуры представления программного кода при статическом анализе.

Решаемые проблемы:

1. Анализ особенностей лексического и синтаксического анализа языков программирования с динамической типизацией, включая проблемы формализации грамматики и построения устойчивых синтаксических структур.
2. Исследование роли абстрактного синтаксического дерева как основы для представления программ и последующего извлечения семантической информации.
3. Рассмотрение методов частичного семантического анализа, применимых в условиях отсутствия статически определённых типов, включая анализ областей видимости, присваиваний и контекста использования идентификаторов.
4. Определение возможностей и ограничений извлечения семантической информации без выполнения программного кода.

Ограничения: рассматривается только статический анализ, без учёта динамического выполнения; полноценный семантический анализ (точное определение типов и поведения) не проводится.

Научная новизна – систематизация подходов к синтаксическому и частичному семантическому анализу динамически типизированных языков и формулировка обобщённой модели анализа на основе AST как универсального промежуточного представления, позволяющего извлекать семантические характеристики и выявлять потенциальные ошибки на этапе статического анализа.

III. Методы исследования

Применялись методы теоретического, структурного и сравнительного анализа. Рассматривались подходы к лексическому и синтаксическому анализу, построению AST. Для выявления особенностей анализа динамически типизированных языков сравнивались классические методы статической обработки и подходы, адаптированные к отсутствию статически определённых типов. Исследование частичного семантического анализа основывалось на структурном анализе AST с учётом контекста идентификаторов. Моделирование позволило сформировать концептуальную модель анализа.

IV. Теория

1. Лексический и синтаксический анализ языков с динамической типизацией.

Лексический анализ преобразует исходный текст в последовательность лексем и не имеет принципиальных отличий от аналогичного этапа для строго

типизированных языков. Синтаксический анализ проверяет соответствие лексем грамматике и строит синтаксическое или абстрактное синтаксическое дерево. При динамической типизации этот этап обеспечивает корректность структуры, но не выявляет семантические характеристики, связанные с типами и контекстом выполнения. Синтаксически корректные программы могут содержать семантические ошибки, обнаруживаемые только в run-time [2]. Поэтому результаты синтаксического анализа служат основой для частичного семантического анализа. Общая последовательность обработки программного кода, включающая рассматриваемые в работе этапы, представлена на рисунке 1.

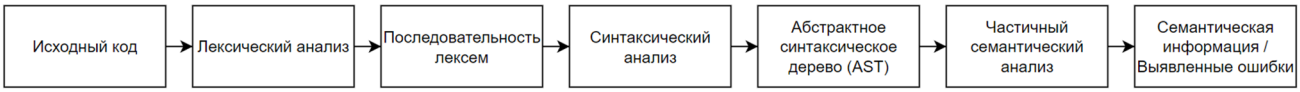

Рис. 1. Блок-схема этапов анализа

2. Абстрактное синтаксическое дерево как модель представления программы.

AST сохраняет только семантически значимые конструкции, исключая избыточные элементы грамматики. Оно представляет программу в виде иерархической структуры, независимой от исходного текста. В динамически типизированных языках AST позволяет извлекать семантическую информацию при отсутствии явных типов: на его основе выполняется анализ областей видимости, последовательностей операций и контекста идентификаторов, что делает AST ключевым промежуточным представлением для частичного семантического анализа [2,3].

Пример построения AST для простого арифметического выражения показан на рисунке 2 и рисунке 3.

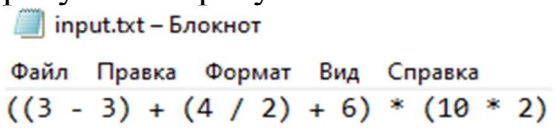

Рис. 2. Выражение

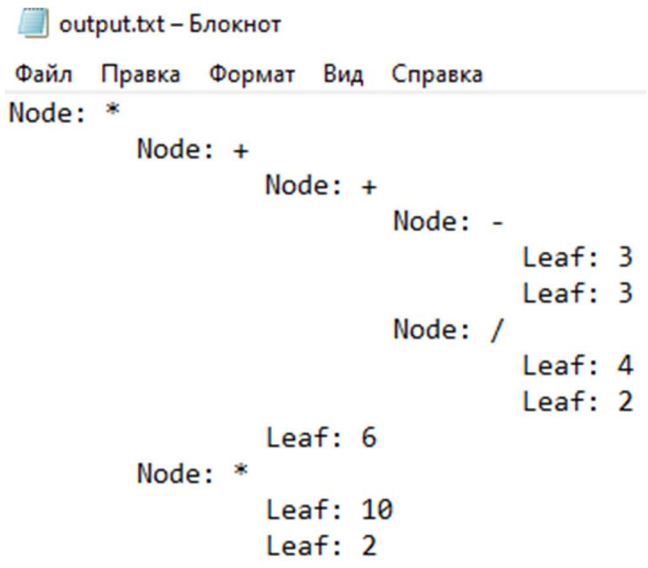

Рис. 3. Абстрактное синтаксическое дерево

Для представленного на рисунке 2 выражения абстрактное синтаксическое дерево построено на рисунке 3. Постфиксный обход этого дерева (левое поддерево → правое поддерево → корень) даёт обратную польскую нотацию (ОПН):

$$33 – 42/ + 6 + 10\ 2** \quad (1)$$

Такое представление удобно для семантического анализа, так как позволяет проверять корректность количества операндов и, с использованием эвристик, оценивать соответствие типов без выполнения программы.

3. Частичный семантический анализ программ в языках с динамической типизацией

Полный статический семантический анализ в динамически типизированных языках затруднён из-за определения типов в run-time. Частичный анализ использует аппроксимации и эвристики, опираясь на AST [3]. Извлекаются данные, не требующие точного знания типов: области видимости идентификаторов, последовательности присваиваний, структура вызовов функций, контекст выражений. Это позволяет выявлять потенциальные ошибки (необъявленные идентификаторы, некорректное переопределение переменных, логические противоречия). Анализ контекстно-зависим: свойства элементов зависят от области видимости и порядка выполнения, что требует иерархических проходов по AST [3]. Несмотря на ограниченность, такой подход повышает качество статической проверки и служит основой для интеллектуальных средств поддержки программирования.

V. Обсуждение результатов

Рассмотренные подходы показывают, что эффективный статический анализ возможен даже для динамически типизированных языков при использовании AST и контекстно-зависимых методов. AST отделяет синтаксис от реализации и служит базой для семантического анализа [3]. Частичный семантический анализ выявляет ошибки, связанные с идентификаторами, областями видимости и логикой, что полезно для инструментов поддержки программирования [4]. Предложенные методы обеспечивают баланс между точностью и вычислительной сложностью.

В таблице 1 приведено сравнение классического статического анализа (ориентированного на строго типизированные языки) и частичного семантического анализа, применяемого для динамически типизированных языков.

Табл. 1. Сравнение подходов к статическому анализу

Критерий	**Классический статический анализ**	**Частичный семантический анализ**
Требования к типам	Явное объявление типов	Типы могут быть неизвестны статически
Объект анализа	Типовая совместимость, сигнатуры функций	Области видимости, необъявленные идентификаторы, контекст использования
Базовая структура	Таблицы символов, графы вызовов	AST, графы потоков данных, эвристики
Примеры инструментов	Компиляторы (gcc, javac)	Линтеры (Pylint, ESLint)
Полнота семантики	Высокая (при строгой типизации)	Частичная, но достаточная для многих задач
Вычислительная сложность	Умеренная	Низкая (благодаря эвристикам)

VI. Выводы и заключение

Проанализированы подходы к синтаксическому и частичному семантическому анализу динамически типизированных языков. Показано, что использование AST в сочетании с контекстно-зависимыми методами позволяет эффективно обрабатывать код без полного знания типов. Основной результат — обоснование применимости частичного семантического анализа для повышения качества кода и раннего обнаружения ошибок. Предложенные подходы могут быть использованы в средствах анализа и редакторах кода для динамически типизированных языков. Дальнейшие исследования могут быть направлены на гибридные методы, сочетающие статический и динамический анализ.

Список литературы

1. Попова В.А. Матричный подход к статической проверке типов в программных продуктах 1С // Экономика строительства: сборник научных трудов. – М.: Изд-во ООО “Русайнс”, 2025. – №11. – С. 539-542.
2. Игнатьев В.Н. Организация статического анализа на абстрактных синтаксических деревьях с помощью конечных автоматов // Труды ИСП РАН. – 2025. – Т. 37, №1. – С. 41-54. – DOI: 10.15514/ISPRAS–2025–37(1)–2.
3. Грибков Н.А., Овасапян Т.Д., Москвин Д.А. Анализ восстановленного программного кода с использованием абстрактных синтаксических деревьев // Проблемы информационной безопасности. Компьютерные системы. – 2023. – №2(54). – С. 47-60. – DOI: 10.48612/jisp/ruar-u6he-kmd4.
4. Романчук К.А. Статический анализ исходного кода на языке typescript // 75-я научная конференция студентов и аспирантов Белорусского государственного университета: Материалы конференции. В 3-х частях, Минск, 14-23 мая 2018 года. Часть 3. – Минск: Белорусский государственный университет, 2018. – С. 560-563.

Сведения об авторе:
Саитов Руслан Ильмирович – магистрант.