DOI: 10.58168/CISMP2024_126-129
УДК 004.9

ЭКВИВАЛЕНТ В МАТЕМАТИКЕ И ИНФОРМАТИКЕ

И.А. Поярков, Е.С. Ильин
ilya.poyarkov776@gmail.com

ФГБОУ ВО «Воронежский государственный лесотехнический университет имени Г.Ф. Морозова»

Аннотация. Эквивалентность в контексте информационных технологий является важной концепцией, лежащей в основе множества аспектов разработки программного обеспечения и теоретической информатики. Настоящая работа рассматривает понятие эквивалентности, её виды и применение в области ИТ. Эквивалентность программных решений, алгоритмов и логических выражений позволяет обеспечить корректность работы программ, оптимизировать их производительность и снизить вероятность ошибок. В статье обсуждаются подходы к проверке эквивалентности программного кода, логической эквивалентности выражений и методов оптимизации эквивалентных решений. Анализируются преимущества использования эквивалентности в программировании, такие как повышение надежности и эффективности программ, а также выявляются трудности, связанные с проверкой полной эквивалентности и оптимизацией алгоритмов. Работа подчеркивает значимость эквивалентности в современных системах разработки программного обеспечения и её роль в улучшении качества программного продукта.

Ключевые слова: эквивалентность, логика, алгоритмы, программирование, теории множеств.

EQUIVALENT IN MATHEMATICS AND COMPUTER SCIENCE

I.A. Poyarkov, E.S. Ilyin
ilya.poyarkov776@gmail.com

Voronezh State University of Forestry and Technologies named after G.F. Morozov

Abstract. Equivalence in the context of information technology is an important concept that underlies many aspects of software development and theoretical computer science. This paper explores the concept of equivalence, its types, and its application in the IT field. Equivalence of software solutions, algorithms, and logical expressions ensures the correct functioning of programs, optimizes their performance, and reduces the likelihood of errors. The article discusses approaches to verifying code equivalence, logical equivalence of expressions, and methods for optimizing equivalent solutions. The advantages of using equivalence in programming, such as improving the reliability and efficiency of programs, are analyzed, as well as the challenges associated with verifying full equivalence and optimizing algorithms. The paper emphasizes the importance of equivalence in modern software development systems and its role in improving software quality.

Keywords: equivalence, logic, algorithms, programming, set theory.

Эквивалентность — это фундаментальное понятие в теории программирования и информатики, которое описывает ситуацию, при которой два объекта (программы, алгоритмы, логические выражения) дают одинаковый результат при одинаковых входных данных. Существует несколько видов эквивалентности, включая синтаксическую эквивалентность (совпадение по структуре), семантическую эквивалентность (совпадение по функциональности) и логическую эквивалентность (равнозначность логических выражений). Эти виды эквивалентности важны для оптимизации программного кода, поскольку позволяют находить альтернативные, но эквивалентные решения для ускорения работы программы или улучшения её структуры [1].

Для проверки эквивалентности применяются разные методы, каждый имеет свои плюсы и минусы. К таким методам относятся: статический анализ кода, тестирование и т.д. Однако задача проверки полной эквивалентности может быть очень трудной, особенно для сложных и объёмных программ. Не зря считают ИТ-специалисты, её актуальной проблемой в ИТ-сфере.

Синтаксическая эквивалентность имеет такое свойство, что два кода или выражения, могут выглядит как две капли воды и быть написаны одними и теми же символами и правилами грамматики языка программирования. Такое свойство научно называется одинарная структура [2].

Например, два следующих выражения в коде:

```
int z = 5;
int v = z + 2;
int z = 5;
int v = 5 + 2;
```

Они синтаксически неэквивалентны. Так как в первом случае используется переменная `z`, а во втором её значение `5`. Однако в случае, если выражения абсолютно идентично схожи по структуре и символам, они будут считаться синтаксически эквивалентными [3].

Преимущества синтаксической эквивалентности:

1. Легкость проверки: Сравнение на уровне символов и структур в коде, намного проще, чем анализ поведения программы.
2. Поддержка инструментов статического анализа. Простой инструмент, который использует синтаксическую эквивалентность, для анализа кода и выявление ошибок.

Недостатки синтаксической эквивалентности:

1. Синтаксически эквивалентные программы не обязательно работают одинаково. Программы могут быть эквивалентны синтаксически, но они отличаться по поведению, если контекст выполнения изменяет их работу. Например, из-за разных значений переменных или условий выполнения, программы будут работать различно.

Логическая эквивалентность — углубленный уровень эквивалентности, чем синтетическая. В логической эквивалентности два выражения или программы, могут иметь разную структуру или быть синтаксически различны, но давать один и тот же результат при любых условиях выполнения. Это очень

важная концепция в программировании и математической логике, особенно, если это касается разработки и программ [5].

Пример логической эквивалентности в программировании:

```
if (i > j) {
   return true;
} else {
   return false;
}
```

И

```
return i > j;
```

Эти два фрагмента синтаксически абсолютные разные, но логически эквивалентны, то есть равны. Они выполняют одну и ту же функцию — возвращают результат сравнения `i` и `j`.

Преимущества логической эквивалентности:

1. Повышает гибкость программирования: Разработчик может писать программу абсолютно разными способами, и гарантировано получит один и тот же результат.
2. Оптимизация программ. Логическая эквивалентность позволят заменить, сложные фрагменты кода, на более простые, получая тот же результат.

Недостатки логической эквивалентности:

1. Трудность проверки заключается в вычислительной сложности задач автоматической проверки логической эквивалентности. Это особо видно на больших, объёмных и сложные программ. В некоторых случая, иногда просто не решаемой.
2. Необходимость формальных методов: требуется для сложных программ, использование формальной верификации, что требует для программиста обширных знаний и значительных усилий [6].

Таким образом, эквивалент представляет собой концепцию, которая описывает равнозначность различных объектов в программировании и информатике, таких как программы, алгоритмы или логические выражения, при условии, что они дают одинаковые результаты на одинаковых входных данных.

Список литературы

1. Касьянов, В. В. Теория алгоритмов и вычислимости : учебное пособие / В. В. Касьянов ; Московский государственный университет. – Москва: МГУ, 2011. – 320 с.
2. Гринченко, С. Н. Формальные методы программирования: учебное пособие / С. Н. Гринченко, С. А. Пузырев; Санкт-Петербургский политехнический университет. – Санкт-Петербург: СПбГПУ, 2006. – 250 с.
3. Левин, И. Б. Алгоритмы и структуры данных: учебное пособие / И. Б. Левин, Е. А. Лившиц; Московский государственный университет. – Москва: МГУ, 2010. – 340 с.

4. Гаврилов, Г. П. Методы оптимизации программ: монография / Г. П. Гаврилов, Ю. И. Розенблюм; Московский институт электроники и математики. – Москва: Радио и связь, 1987. – 210 с.

5. Ильин, И. В. Программирование: синтаксис и семантика языков: учебное пособие / И. В. Ильин; Московский государственный технический университет. – Москва: Бином, 2009. – 312 с.

6. Мищенко, С. Г. Формальные языки и грамматики в программировании / С. Г. Мищенко, А. Н. Шевцов; Московский государственный технический университет. –Москва:МГТУ им. Н. Э. Баумана, 2012. – 230 с.

7. Скрыпников А.В., Стукало О.Г., Денисенко В.В., Бакаев Д.Н., Минакова А.А., Бутенко А.О. Моделирование рисков реализации проектов развития предприятий мясоперерабатывающей промышленности // Моделирование систем и процессов. – 2022. – Т. 15, № 3. – С. 77-83.

References

1. Kasyanov, V. V. Theory of Algorithms and Computability: A Textbook / V. V. Kasyanov; Moscow State University. – Moscow: MSU, 2011. – 320 p.

2. Grinchenko, S. N. Formal Methods of Programming: A Textbook / S. N. Grinchenko, S. A. Puzyrev ; Saint Petersburg Polytechnic University. – Saint Petersburg: SPbPU, 2006. – 250 p.

3. Levin, I. B. Algorithms and Data Structures: A Textbook / I. B. Levin, E. A. Livshits; Moscow State University. – Moscow: MSU, 2010. – 340 p.

4. Gavrilov, G. P. Methods of Program Optimization: A Monograph / G. P. Gavrilov, Yu. I. Rozenblyum ; Moscow Institute of Electronics and Mathematics. – Moscow: Radio i Svyaz, 1987. – 210 p.

5. Ilyin, I. V. Programming: Syntax and Semantics of Languages: A Textbook / I. V. Ilyin; Moscow State Technical University. – Moscow: Binom, 2009. – 312 p.

6. Mishchenko, S. G. Formal Languages and Grammars in Programming / S. G. Mishchenko, A. N. Shevtsov; Moscow State Technical University. – Moscow: Bauman Moscow State Technical University, 2012. – 230 p.

7. Skrypnikov, A.V.; Stukalo, O.G.; Denisenko, V.V.; Bakaev, D.N.; Minakova, A.A.; Butenko, A.O. Modeling of the risks of implementation of development projects of meat processing enterprises // Modeling of systems and processes. - 2022. - Т. 15, № 3. - С. 77-83.