TURING MASHINASI VA HISOBLASH TO'LIQLIGI KONSEPSIYASINING ZAMONAVIY DASTURLASH TILLARIDA NAMOYISHI

Jovliyeva Dilnoz Mustofa qizi
Xalqaro innovatsion universiteti o‘qituvchisi
dilziyoo@gmail.com
ORCID-0009-0009-6123-172X
https://doi.org/10.5281/zenodo.20085816

Annotatsiya: Ushbu maqolada Turing mashinasi konsepsiyasi va hisoblash to'liqligi tushunchalarining zamonaviy dasturlash tillarida qanday namoyon bo'lishi nazariy jihatdan tahlil qilingan. Tadqiqotda Alan Turing tomonidan 1936-yilda taqdim etilgan universal hisoblash modeli va uning zamonaviy dasturlash amaliyotidagi ahamiyati ko'rib chiqiladi. Maqolada Turing to'liqligi mezonining Python, Java, C++ va boshqa keng tarqalgan dasturlash tillarida qanday ta'minlanishi adabiyotlar tahlili orqali baholanadi. Asosiy e'tibor nazariy informatika konsepsiyalarining amaliy dasturlashga ko'prik vazifasini bajarishiga qaratilgan.

Kalit so'zlar: Turing mashinasi, hisoblash to'liqligi, dasturlash tillari, formal tillar nazariyasi, lambda-hisob, Church-Turing tezisi, universal hisoblash modeli.

Аннотация. В данной статье проводится теоретический анализ проявления концепции машины Тьюринга и понятия полноты по Тьюрингу в современных языках программирования. Исследование рассматривает универсальную вычислительную модель, представленную Аланом Тьюрингом в 1936 году, и её значение в современной практике программирования. В статье оценивается реализация критерия полноты по Тьюрингу в Python, Java, C++ и других широко распространённых языках программирования посредством анализа литературы. Основное внимание уделяется роли концепций теоретической информатики как моста к практическому программированию.

Ключевые слова: машина Тьюринга, полнота по Тьюрингу, языки программирования, теория формальных языков, лямбда-исчисление, тезис Чёрча-Тьюринга, универсальная вычислительная модель.

Abstract. This article provides a theoretical analysis of how the Turing machine concept and Turing completeness notion manifest in modern programming languages. The study examines the universal computational model introduced by Alan Turing in 1936 and its significance in contemporary programming practice. The article evaluates the implementation of Turing completeness criteria in Python, Java, C++ and other widely used programming languages through literature analysis. Primary focus is placed on the role of theoretical computer science concepts as a bridge to practical programming.

Keywords: Turing machine, Turing completeness, programming languages, formal language theory, lambda calculus, Church-Turing thesis, universal computational model.

Kirish. Informatika nazariyasining rivojlanishida Alan Turing 1936-yilda taqdim etgan abstrakt hisoblash modeli muhim asos bo'lib xizmat qiladi. Turing mashinasi deb nomlanuvchi ushbu model har qanday algoritm orqali hal qilinadigan masalaning matematik formalizmini taqdim etadi va zamonaviy hisoblash nazariyasining poydevori hisoblanadi [1]. Turing to'liqligi

konsepsiyasi esa har qanday dasturlash tili yoki hisoblash tizimining nazariy jihatdan Turing mashinasi bilan ekvivalent hisoblash quvvatiga ega ekanligini bildiradi. Ushbu konsepsiya nazariy informatikadan amaliy dasturlashga o'tishda muhim ko'prik vazifasini bajaradi, chunki u dasturlash tillarining universal hisoblash qobiliyatini baholash uchun aniq mezon beradi [2]. Zamonaviy dasturlash tillari ko'plab sintaktik va semantik farqlarga ega bo'lishiga qaramay, ularning aksariyati Turing to'liqligi mezoniga javob beradi, ya'ni nazariy jihatdan bir-birlari bilan ekvivalent hisoblash quvvatiga ega. Biroq, bu nazariy ekvivalentlik amaliy dasturlashda turli xil ifoda vositalari va abstraksiya darajalarida namoyon bo'ladi. Ushbu tadqiqotning maqsadi Turing mashinasi va hisoblash to'liqligi konsepsiyalarining zamonaviy dasturlash tillarida qanday namoyon bo'lishini adabiyotlar tahlili va nazariy baholash orqali o'rganishdir.

Metodologiya va adabiyotlar tahlili. Ushbu tadqiqotda qo'llanilgan metodologiya asosan nazariy tahlil va adabiyotlar tahlili usullariga asoslanadi. Turing mashinasining tuzilishi va ishlash tamoyillari Hopcroft va Ullman tomonidan yaratilgan klassik asarlar asosida ko'rib chiqildi [3]. Sipser tomonidan taqdim etilgan zamonaviy hisoblash nazariyasi yondashuvlari orqali Turing to'liqligi mezonining aniq ta'rifi va uning ahamiyati baholandi [4]. Church-Turing tezisi lambda-hisob va Turing mashinasi o'rtasidagi ekvivalentlikni ta'kidlaydi, bu esa har qanday effektiv hisoblash usulining ushbu ikkala formalizmda ifoda etilishi mumkinligini ko'rsatadi [5]. Zamonaviy dasturlash tillari kontekstida, Sebesta tomonidan taqdim etilgan dasturlash tillari konsepsiyalari bo'yicha fundamental asarda imperativ, funksional va ob'ektga yo'naltirilgan paradigmalarning hisoblash quvvati tahlil qilingan [6]. Python dasturlash tili bo'yicha rasmiy hujjatlar va akademik tadqiqotlar ushbu tilning Turing to'liqligi Python Virtual Machine mexanizmi orqali ta'minlanishini ko'rsatadi, bu erda dinamik tipizatsiya va rekursiya qo'llab-quvvatlanadi [7].

Java dasturlash tilining hisoblash to'liqligi Gosling va boshqalar tomonidan til spetsifikatsiyasida ta'kidlangan bo'lib, Java Virtual Machine arxitekturasi Turing mashinasi modeliga to'liq mos keladi [8]. C++ tilining Turing to'liqligi Stroustrup tomonidan til dizaynida nazarda tutilgan bo'lib, shablonlar mexanizmi orqali hatto kompilyatsiya vaqtida ham to'liq hisoblash quvvati ta'minlanadi [9]. Funksional dasturlash paradigmasi kontekstida Haskell kabi tillar lambda-hisob asosida qurilganligi sababli tabiiy ravishda Turing to'liq hisoblanadi, bu Pierce tomonidan funksional dasturlash nazariyasi asosida ko'rsatilgan [10]. Adabiyotlar tahlili shuni ko'rsatadiki, Turing to'liqligi uchun dasturlash tilida uchta asosiy xususiyat mavjud bo'lishi zarur: birinchidan, ixtiyoriy hajmdagi xotirani boshqarish qobiliyati; ikkinchidan, shartli tarmoqlanish va takrorlash konstruksiyalari; uchinchidan, rekursiya yoki sikl orqali cheksiz hisoblashlarni ifoda etish imkoniyati. Ushbu mezonlar zamonaviy dasturlash tillarining deyarli barchasida to'liq qo'llab-quvvatlanadi, ammo ularning sintaktik va semantik ifodalanishi turlicha bo'ladi.

Natijalar va muhokama. Adabiyotlar tahlili va nazariy baholash natijalari shuni ko'rsatadiki, Turing mashinasi konsepsiyasi zamonaviy dasturlash tillarida abstrakt tarzda namoyon bo'ladi, lekin uning asosiy tamoyillari saqlanib qoladi. Turing mashinasining asosiy komponentlari - cheksiz lenta, bosh, holat o'tish funksiyasi - zamonaviy dasturlash tillarida xotira, ko'rsatkichlar yoki indekslar, va boshqaruv oqimi konstruksiyalari shaklida mavjud. Python dasturlash tilida Turing to'liqligi ro'yxatlar va lug'atlar orqali dinamik xotira boshqaruvi, while va for sikllar orqali takrorlash, if-elif-else orqali shartli tarmoqlanish, va def kalit so'zi orqali rekursiv funksiyalar yordamida ta'minlanadi. Masalan, har qanday Turing

mashinasini Python dasturida simulyatsiya qilish mumkin, bu esa tilning nazariy to'liqligi uchun amaliy dalil beradi. Java tilida esa ob'ektga yo'naltirilgan yondashuv Turing mashinasi holatlarini sinflar va metodlar orqali modellashtirish imkonini beradi, bunda ArrayList yoki HashMap kabi to'plamlar cheksiz lenta vazifasini bajaradi. C++ tilida shablonlar mexanizmi juda qiziqarli xususiyat bo'lib, kompilyatsiya vaqtida Turing to'liq hisoblashlarni amalga oshirish mumkin, bu Template Metaprogramming deb ataluvchi texnikada namoyon bo'ladi.

Funksional dasturlash tillari, xususan Haskell va Lisp, lambda-hisob to'g'ridan-to'g'ri ifoda etilgani uchun Church-Turing tezisi doirasida tabiiy ravishda Turing to'liq hisoblanadi. Biroq, muhokama qilish zarur bo'lgan muhim nuqta shundaki, nazariy Turing to'liqligi va amaliy dasturlash o'rtasida farq mavjud. Nazariy jihatdan cheksiz xotira va cheksiz vaqt taxmin qilinsa ham, real dasturlash muhitlarida xotira va vaqt resurslari cheklangan. Shuningdek, ba'zi dasturlash tillari ataylab Turing to'liq bo'lmaslik uchun loyihalashtirilgan, masalan, HTML yoki SQL kabi deklarativ tillar ma'lum bir domenga yo'naltirilgan va umumiy maqsadli hisoblashlar uchun mo'ljallanmagan. Turing to'liqligi konsepsiyasining amaliy ahamiyati shundaki, u dasturlash tillarining nazariy quvvatini baholash uchun universal standart beradi va tillar o'rtasida o'tish yoki yangi til yaratishda asosiy mezonlarni aniqlashga yordam beradi. Zamonaviy dasturlash paradigmalari - imperativ, funksional, ob'ektga yo'naltirilgan, mantiqiy - turli sintaktik shakllarda bo'lishiga qaramay, hisoblash quvvati jihatidan ekvivalentdir, chunki ularning barchasi Turing to'liqligi mezoniga javob beradi. Natijada, dasturchi ma'lum bir paradigma yoki tilni tanlashda sintaktik qulaylik, ishlash tezligi, xotira boshqaruvi kabi amaliy omillarni hisobga olishi mumkin, chunki nazariy jihatdan barcha Turing to'liq tillar bir xil masalalarni hal qilish qobiliyatiga ega.

Xulosa. Ushbu tadqiqot Turing mashinasi va hisoblash to'liqligi konsepsiyalarining zamonaviy dasturlash tillarida qanday namoyon bo'lishini nazariy jihatdan o'rganishga bag'ishlandi. Adabiyotlar tahlili va nazariy baholash natijalariga ko'ra, quyidagi xulosalarga kelindi: birinchidan, zamonaviy dasturlash tillarining aksariyati Turing to'liqligi mezoniga javob beradi va nazariy jihatdan bir xil hisoblash quvvatiga ega; ikkinchidan, Turing to'liqligi uchun zarur bo'lgan asosiy xususiyatlar - xotira boshqaruvi, shartli tarmoqlanish va rekursiya - turli sintaktik shakllarda bo'lishiga qaramay, barcha zamonaviy tillarda mavjud; uchinchidan, nazariy Turing to'liqligi va amaliy dasturlash o'rtasida resurslar cheklanishi tufayli farq mavjud. Tadqiqot natijalarining nazariy ahamiyati shundaki, u informatika nazariyasi va amaliy dasturlash o'rtasidagi bog'lanishni aniqlashtiradi va dasturlash tillarini baholash uchun universal mezon beradi. Amaliy jihatdan esa, tadqiqot natijalari dasturchilar va informatika mutaxassislari uchun turli tillarda ishlashda nazariy asoslarni tushunish va yangi texnologiyalarni qabul qilishda foydali bo'lishi mumkin.

Adabiyotlar, References, Литературы:

1. Turing A.M. On Computable Numbers, with an Application to the Entscheidungsproblem // Proceedings of the London Mathematical Society. – 1936. – Vol. 42. – P. 230-265.
2. Дэвис М. Вычислимость и неразрешимость. – М.: Мир, 1978. – 232 с.
3. Hopcroft J.E., Ullman J.D. Introduction to Automata Theory, Languages, and Computation. – Reading: Addison-Wesley, 1979. – 418 p.
4. Sipser M. Introduction to the Theory of Computation. – 3rd ed. – Boston: Cengage Learning, 2012. – 504 p.

5. Church A. An Unsolvable Problem of Elementary Number Theory // American Journal of Mathematics. – 1936. – Vol. 58. – No. 2. – P. 345-363.
6. Sebesta R.W. Concepts of Programming Languages. – 11th ed. – Boston: Pearson, 2015. – 792 p.
7. Van Rossum G., Drake F.L. The Python Language Reference Manual. – Bristol: Network Theory Ltd., 2011. – 151 p.
8. Gosling J., Joy B., Steele G., Bracha G. The Java Language Specification. – 3rd ed. – Boston: Addison-Wesley, 2005. – 688 p.
9. Stroustrup B. The C++ Programming Language. – 4th ed. – Upper Saddle River: Addison-Wesley, 2013. – 1346 p.
10. Pierce B.C. Types and Programming Languages. – Cambridge: MIT Press, 2002. – 623 p.