Taraqqiyot spektri
ILMIY JURNALI
Vol 2 № 11 2026

ZAMONAVIY DASTURLASH TILLARINING SAMARADORLIGINI QIYOSIY TAHLIL QILISH: PYTHON, C++ VA JAVA MISOLIDA

Taxirova Sohiba Alisherovna

OXUS University kompyuter ilimlari 501-26 guruh talabasi 100105, O'zbekiston

oxusuniversiteti@gmail.com

Tursunboyev Xaydarali To'lqin o'g'li

OXUS University kompyuter ilimlari 501-26 guruh talabasi 111500, O'zbekiston

xaydaralitursunboyev1@gmail.com

Samandar Yusupov Anvarovich

OXUS Universiteti katta o'qituvchisi 100208, O'zbekiston

Samandar75781111@gmail.com

Annotatsiya: Mazkur maqolada zamonaviy dasturiy ta'minot ishlab chiqishda keng qo'llaniladigan Python, C++ va Java dasturlash tillarining samaradorligi qiyosiy jihatdan tahlil qilinadi. Tadqiqotda samaradorlik tushunchasi faqat bajarilish tezligi bilan cheklanmay, xotiradan foydalanish, ishlab chiqish qulayligi, platformalararo moslik, kutubxonalar ekotizimi, kengayuvchanlik va qo'llash sohasi kabi mezonlar orqali ko'rib chiqiladi. Har bir tilning arxitekturaviy xususiyatlari, bajarilish modeli va amaliy afzalliklari tahlil qilinib, ularni tanlash bo'yicha tavsiyalar ishlab chiqiladi. Maqolada taklif etilgan metodologiya bir xil algoritmik masalalarni uchala tilda amalga oshirish, bir xil apparat va dasturiy muhitda sinash hamda natijalarni statistik mezonlar asosida baholashga asoslanadi.

Hozirgi maqolada real benchmark tajribasi o‘tkazilgani da’vo qilinmaydi; natijalar bo‘limida adabiyotlar va tillarning texnik xususiyatlariga tayangan sifat jihatidan qiyosiy tahlil beriladi.

Kalit so‘zlar: dasturlash tillari, Python, C++, Java, samaradorlik, bajarilish vaqti, xotira sarfi, algoritm, qiyosiy tahlil, dasturiy ta’minot.

Kirish: Raqamli iqtisodiyot, avtomatlashtirish va axborot tizimlarining jadal rivojlanishi dasturiy ta’minotga bo‘lgan talabni muntazam oshirmoqda. Bugungi kunda dasturlash tillari veb-ilovalar, mobil tizimlar, sanoat boshqaruvi, ilmiy hisoblashlar, sun’iy intellekt va katta hajmdagi ma’lumotlarni qayta ishlash kabi ko‘plab yo‘nalishlarda qo‘llaniladi. Dasturiy mahsulotning sifati esa tanlangan dasturlash tili, algoritm, dastur arxitekturasi va bajarilish muhitiga bevosita bog‘liq.

Python, C++ va Java zamonaviy dasturlash amaliyotida muhim o‘rin egallagan tillardir. Python sodda sintaksisi va boy kutubxonalar ekotizimi bilan tezkor prototiplash, ma’lumotlar tahlili hamda sun’iy intellekt loyihalarida keng qo‘llanadi. C++ apparat resurslariga yaqin ishlash, xotirani boshqarish imkoniyati va yuqori unumdorligi sababli o‘yin dvijoklari, real vaqt tizimlari va resurs talab qiluvchi dasturlarda muhim hisoblanadi. Java esa virtual mashina, avtomatik xotira boshqaruvi va platformalararo ishlash imkoniyatlari tufayli korporativ axborot tizimlari hamda server ilovalarida keng tarqalgan.

Ushbu tillarni “qaysi biri eng yaxshi?” degan yagona savol bilan baholash to‘g‘ri emas. Chunki samaradorlik loyihaning maqsadi, ma’lumotlar hajmi, algoritm murakkabligi, ishga tushirish muhiti va ishlab chiqish muddatiga qarab o‘zgaradi. Masalan, hisoblash tezligi ustuvor bo‘lgan vazifada C++ afzal bo‘lishi mumkin, biroq dasturiy prototipni tez yaratish talab etilsa Python maqsadga muvofiq bo‘ladi. Java esa katta tizimlarda barqarorlik, boshqariluvchanlik va platformalararo muvofiqlikni ta’minlashi mumkin.

Tadqiqotning maqsadi – Python, C++ va Java dasturlash tillarining texnik hamda amaliy samaradorligini umumiy mezonlar asosida qiyoslash, ularning kuchli va cheklangan

jihatlarini aniqlash hamda muayyan vazifalar uchun til tanlash bo‘yicha ilmiy-amaliy tavsiyalar ishlab chiqishdan iborat.

• **Tadqiqot vazifalari:** samaradorlik mezonlarini belgilash; uchala tilning bajarilish modeli va resurslardan foydalanishini tavsiflash; mavjud ilmiy adabiyotlarni tahlil qilish; takrorlanuvchi tajriba metodikasini ishlab chiqish; natijalarni qo‘llash sohasi bo‘yicha umumlashtirish.

• **Tadqiqot obyekti:** Python, C++ va Java tillarida yaratilgan dasturiy yechimlar.

• **Tadqiqot predmeti:** dasturlash tillarining bajarilish tezligi, xotira sarfi, ishlab chiqish samaradorligi va amaliy qo‘llash imkoniyatlari.

Mavzuga doir adabiyotlar tasnifi: Dasturlash tillarining samaradorligini baholash bo‘yicha ilmiy tadqiqotlarda turli yondashuvlar mavjud. Prechelt tomonidan o‘tkazilgan empirik taqqoslashda bir xil masalani turli dasturlash tillarida yechish natijalari, jumladan, dastur hajmi, ishlab chiqish vaqti va bajarilish xususiyatlari o‘rganilgan. Ushbu ish dasturlash tillarini faqat sintaksis yoki nazariy imkoniyatlar asosida emas, balki bir xil topshiriqdagi amaliy natijalar orqali baholash muhimligini ko‘rsatadi.

Pereira va hammualliflarining tadqiqotida dasturlash tillarining energiya sarfi, bajarilish vaqti va xotira iste’moli o‘zaro bog‘liq holda tahlil qilingan. Tadqiqot natijalari unumdorlikni baholashda bitta ko‘rsatkich bilan cheklanmaslik, energiya va xotira kabi resurslarni ham hisobga olish zarurligini asoslaydi.

Python rasmiy hujjatlarida tilning yuqori darajali ma’lumot tuzilmalari, dinamik tiplash va keng standart kutubxonasi dastur ishlab chiqishni soddalashtirishi ta’kidlanadi. Shu bilan birga, odatiy CPython bajarilish muhitida bayt-kod interpretatsiyasi va obyektlar bilan ishlash qo‘shimcha xarajat keltirib chiqarishi mumkin. Biroq NumPy, pandas va boshqa optimallashtirilgan kutubxonalar hisoblashning katta qismini C yoki Fortran kabi kompilyatsiyalanuvchi tillarda bajarib, Python interfeysi orqali yuqori samaradorlikka erishish imkonini beradi.

C++ tilining asosiy xususiyatlari – kompilyatsiya qilinishi, statik tiplash, xotira va resurslarni boshqarish imkoniyatlarining kengligi hamda apparatga yaqin ishlashidir. C++ standarti tilning umumiy qoidalarini belgilaydi, biroq amaliy tezlik kompilyator, optimallashtirish parametrlari, standart kutubxona va dastur algoritmiga ham bog‘liq. Demak, C++ning yuqori unumdorlik salohiyati undan to‘g‘ri foydalanish va resurslarni oqilona boshqarishni talab qiladi.

Java platformasida dastur odatda bayt-kodga kompilyatsiya qilinadi va Java Virtual Machine (JVM) muhitida bajariladi. JVMning Just-In-Time (JIT) kompilyatsiya mexanizmi tez-tez bajariladigan kod qismlarini optimallashtirishi mumkin. Java avtomatik xotira boshqaruvi (garbage collection) orqali xotira bilan ishlashni yengillashtiradi, biroq yig‘ish jarayoni va virtual mashina xarajatlari ayrim vaziyatlarda kechikish yoki qo‘shimcha resurs sarfini keltirib chiqarishi mumkin.

Adabiyotlar tahlili shuni ko‘rsatadiki, dasturlash tillarining samaradorligi vazifa va muhitga bog‘liq ko‘p omilli tushunchadir. Shu sababli ushbu maqolada bajarilish vaqti bilan bir qatorda xotira sarfi, ishlab chiqish qulayligi, portativlik va qo‘llash sohasi ham tahlil mezonlari sifatida qabul qilinadi.

Tadqiqot metodologiyasi: Tadqiqot metodologiyasi qiyosiy tahlil, ilmiy adabiyotlarni o‘rganish va nazorat qilinadigan benchmark tajribasini tashkil etish tamoyillariga asoslanadi. Natijalarni xolis taqqoslash uchun uchala dasturlash tilida bir xil algoritm, bir xil kirish ma’lumotlari va imkon qadar bir xil apparat konfiguratsiyasidan foydalanish lozim. Tajriba davomida operatsion tizim, protsessor, tezkor xotira, kompilyator yoki interpretator versiyasi, JVM sozlamalari va optimallashtirish parametrlari qayd etiladi.

1. Samaradorlikni baholash mezonlari

• **Bajarilish vaqti:** dastur belgilangan vazifani bajarishi uchun sarflagan vaqt. Vaqt bir xil o‘lchov birligida, bir necha marta takrorlangan sinovlar asosida o‘lchanadi.

• **Xotira sarfi:** dastur ishlashi davomida egallagan operativ xotiraning maksimal yoki o‘rtacha miqdori. Xotira ko‘rsatkichi bir xil vosita yordamida o‘lchanishi kerak.

• **CPU yuklanishi:** hisoblash jarayonida protsessor resurslaridan foydalanish darajasi. Bu ko‘rsatkich ko‘p oqimli va hisoblashga yo‘naltirilgan vazifalarda ahamiyatli.

• **Ishlab chiqish qulayligi:** kod hajmi, sintaksis murakkabligi, xatolarni aniqlash va tuzatish qulayligi hamda dasturchi mehnatiga ketadigan vaqt.

• **Portativlik va ekotizim:** dasturni turli operatsion tizimlarda ishga tushirish, tayyor kutubxonalar, ishlab chiqish vositalari va hamjamiyat ko‘magining mavjudligi.

2. Tajriba topshiriqlarini tanlash

Amaliy benchmark uchun uch turdagi topshiriq tavsiya etiladi: katta massiv elementlari yig‘indisi va saralash kabi CPUga bog‘liq hisoblash; katta hajmdagi matn yoki fayllarni qayta ishlash; ko‘p sonli obyekt yaratish va ulardan foydalanish orqali xotira boshqaruvini baholash. Har bir topshiriqning algoritmi barcha tillarda bir xil bo‘lishi, kirish ma’lumotlari esa oldindan tayyorlanishi kerak.

3. O‘lchash va hisoblash tartibi

Har bir test kamida 10 marta takrorlanib, dastlabki ishga tushirishdagi kompilyatsiya yoki JVM qizish (warm-up) ta’siri alohida hisobga olinadi. Bajarilish vaqtining medianasi va o‘rtacha qiymati qayd etiladi. Nisbiy tezlikni aniqlash uchun quyidagi ifodadan foydalanish mumkin:

Tezlik koeffitsiyenti = T_r / T_t

Bu yerda T_r – tanlangan tayanch til yoki dastur bajarilish vaqti, T_t – taqqoslanayotgan tilning bajarilish vaqti. Ko‘rsatkich 1 dan katta bo‘lsa, taqqoslanayotgan dastur tayanch variantga nisbatan tezroq bajarilganini bildiradi. Taqqoslashda tayanch til oldindan aniq ko‘rsatilishi zarur.

Xotira sarfi bo‘yicha har bir sinovda maksimal rezident xotira yoki bir xil usulda o‘lchangan xotira ko‘rsatkichi qayd etiladi. Kod hajmi esa satrlar soni bilan birga

funksiyalar, modullar va tashqi bog‘liqliklar murakkabligi nuqtayi nazaridan baholanadi. Olingan raqamlar jadval va diagrammalar orqali taqdim etilishi mumkin.

Tahlil va natijalar: Python, C++ va Java tillarining texnik xususiyatlari bo‘yicha qiyosiy tahlili shuni ko‘rsatadiki, har bir til ma’lum vazifalarda ustunlikka ega. Quyidagi jadvalda tillarning umumiy xususiyatlari sifat jihatidan taqqoslangan. Jadvaldagi baholar universal benchmark natijasi emas; ular tillarning arxitekturasi va odatiy qo‘llanish tajribasiga asoslangan umumlashtirilgan tavsifdir.

Mezon	Python	C++	Java
Bajarilish tezligi	Odatda pastroq; optimallashtirilgan kutubxonalar bilan yuqori bo‘lishi mumkin	Odatda juda yuqori	Yuqori; JIT optimallashtirishiga bog‘liq
Xotira boshqaruvi	Avtomatik, obyektlar xarajati mavjud	Qo‘lda va avtomatik usullar; nozik nazorat	Garbage collection orqali avtomatik
Ishlab chiqish tezligi	Juda qulay, kod ixcham	Nisbatan murakkabroq	Tuzilmali, o‘rta darajada murakkab
Platformalararo ishlash	Interpretator mavjud platformalarda ishlaydi	Har platforma uchun mos kompilyatsiya talab qilishi mumkin	JVM mavjud platformalarda keng moslik
Asosiy qo‘llanishi	AI, data science, avtomatlashtirish, web	O‘yinlar, tizimli dasturlash, real vaqt	Korporativ tizimlar, backend, Android ekotizimi
Asosiy cheklovi	CPUga bog‘liq vazifalarda	Xotira va resurslarni boshqarish	JVM va garbage

Mezon	Python	C++	Java
	interpretator xarajati	murakkabligi	collection xarajatlari

1. Python dasturlash tilining samaradorligi:

Pythonning asosiy ustunligi dasturiy yechimni qisqa vaqt ichida ishlab chiqish imkoniyatidir. Sintaksisining soddaligi, dinamik tiplash va keng kutubxonalar dasturchiga muammoning mantiqiy yechimiga ko‘proq e’tibor qaratish imkonini beradi. Shu sababli prototiplash, ma’lumotlarni tahlil qilish, mashinaviy o‘qitish va avtomatlashtirishda Python samarali tanlov hisoblanadi.

Biroq sof Python kodida katta hajmdagi takroriy hisoblashlar bajarilganda interpretator va obyektlar bilan ishlash xarajatlari sezilishi mumkin. Bunday holatda algoritmni optimallashtirish, vektorlashtirilgan kutubxonalardan foydalanish yoki hisoblash talab qiluvchi qismlarni C/C++ kengaytmalari orqali bajarish samaradorlikni oshiradi. Demak, Pythonning amaliy samaradorligi ko‘pincha uning kutubxonalar ekotizimi va boshqa tillar bilan integratsiyasiga bog‘liq.

2. C++ dasturlash tilining samaradorligi:

C++ kompilyatsiyalanuvchi til sifatida dastur kodini mashina kodiga yaqin shaklda bajarish imkonini beradi. Statik tiplash, xotira ustidan nazorat, ko‘rsatkichlar, ma’lumotlar tuzilmalari va resurslarni boshqarish mexanizmlari yuqori unumdor dasturlar yaratishda muhim ahamiyatga ega. Shu sababli C++ operatsion tizim komponentlari, o‘yin dvijoklari, grafik ishlov berish, simulyatsiya va real vaqtga yaqin tizimlarda keng qo‘llanadi.

C++ning yuqori samaradorlik salohiyati bilan birga, dasturchidan xotira va resurslarni ehtiyotkor boshqarishni talab qilishi uning murakkab tomonidir. Noto‘g‘ri xotira boshqaruvi, resurs oqishi yoki aniqlanmagan xatti-harakatlar dastur barqarorligiga salbiy ta’sir ko‘rsatishi mumkin. Zamonaviy C++ standartlari va smart pointer kabi vositalar bu xavflarni kamaytiradi, ammo dastur arxitekturasi va testlash madaniyati muhimligicha qoladi.

3. Java dasturlash tilining samaradorligi:

Java virtual mashina asosida ishlashi tufayli platformalararo moslikni ta’minlaydi. Dastur bayt-kod shaklida kompilyatsiya qilinadi va JVM mavjud muhitlarda bajariladi. JIT kompilyator tez-tez bajariladigan kod qismlarini ish vaqtida optimallashtirishi mumkin. Bu xususiyat Java dasturlarining uzoq ishlaydigan server jarayonlarida yaxshi unumdorlik ko‘rsatishiga yordam beradi.

Java avtomatik xotira boshqaruvi dasturchining xotirani qo‘lda bo‘shatish zaruratini kamaytiradi. Biroq garbage collection ishlashi, JVMni ishga tushirish va obyektlar modeli qo‘shimcha xotira hamda protsessor xarajatlarini yuzaga keltirishi mumkin. Java katta hajmdagi korporativ tizimlar, mikroxizmatlar va backend platformalarida keng qo‘llanadi; bunda uning kutubxonalari, vositalari va barqaror ekotizimi muhim afzallik hisoblanadi.

4. Umumiy qiyosiy xulosa:

Qiyosiy tahlil asosida C++ resurslarni nozik boshqarish va maksimal bajarilish unumdorligi talab etiladigan vazifalarda kuchli nomzod ekani ko‘rinadi. Python ishlab chiqish tezligi, soddalik va ilmiy kutubxonalar bilan ishlashda ustunlikka ega. Java esa portativlik, katta tizimlarni boshqarish, avtomatik xotira boshqaruvi va server ekotizimi bo‘yicha muvozanatli yechim taklif etadi.

Shu bilan birga, tilning o‘zi yakuniy samaradorlikni belgilamaydi. Noto‘g‘ri algoritm bilan yozilgan C++ dasturi yaxshi optimallashtirilgan Python kutubxonasidan sekin ishlashi mumkin. Xuddi shuningdek, JVMning qizish bosqichi hisobga olinmagan Java benchmarki noto‘g‘ri xulosa berishi mumkin. Shuning uchun adolatli taqqoslashda algoritm, kompilyator/interpreter versiyasi, ishga tushirish parametrlari va test muhiti qat’iy nazorat qilinishi zarur.

Muhim metodologik izoh: ushbu maqola uchun uchala tilda amaliy dasturlar ishga tushirilib, sekund yoki megabayt ko‘rinishida o‘lchangan yangi tajriba natijalari taqdim etilmadi. Shu sababli maqolada aniq foizli ustunlik yoki bajarilish vaqti ko‘rsatkichlari keltirilmagan. Yuqoridagi jadval sifat tahlilidir. Maqolani empirik tadqiqot darajasiga olib

chiqish uchun metodologiyada bayon qilingan benchmark sinovlarini amalda bajarib, haqiqiy o‘lchovlarni jadvalga kiritish lozim.

Xulosa: O‘tkazilgan nazariy va qiyosiy tahlil Python, C++ va Java dasturlash tillarining samaradorligi turli mezonlar bo‘yicha farqlanishini ko‘rsatdi. Python dasturchining mehnatini yengillashtirishi, kodni tez yozish va ma’lumotlar bilan ishlash imkoniyatlari bilan ajralib turadi. C++ yuqori bajarilish tezligi va xotira resurslarini boshqarish imkoniyatlari bilan resurs talab qiluvchi tizimlarda muhim o‘rin egallaydi. Java esa JVM, avtomatik xotira boshqaruvi va keng platformalararo qo‘llab-quvvatlash orqali yirik dasturiy tizimlar uchun qulay muhit yaratadi.

Tadqiqotning asosiy xulosasi shundan iboratki, barcha vazifalar uchun yagona eng samarali dasturlash tili mavjud emas. Til tanlashda loyiha talablari, bajarilish tezligi, xotira cheklovlari, ishlab chiqish muddati, jamoa tajribasi va keyingi qo‘llab-quvvatlash xarajatlari birgalikda hisobga olinishi kerak. Shuningdek, benchmark natijalari faqat aniq apparat va dasturiy konfiguratsiya doirasida talqin qilinishi lozim.

• **Ilmiy ahamiyati:** maqolada samaradorlikni ko‘p mezonli baholash yondashuvi asoslanib, dasturlash tillarini faqat bajarilish tezligi bilan emas, balki xotira, portativlik va ishlab chiqish qulayligi bilan birgalikda tahlil qilish zarurligi ko‘rsatildi.

• **Amaliy ahamiyati:** olingan umumlashtirishlar dasturchilar, talabalar va loyiha rahbarlariga dasturiy mahsulot talablariga mos tilni tanlashda yordam beradi.

Takliflar va keyingi tadqiqot yo‘nalishlari:

• **1.** Python, C++ va Java tillarida bir xil algoritmlarni amalga oshirib, yagona apparat muhitida kamida 10 martadan benchmark sinovlarini o‘tkazish.

• **2.** Bajarilish vaqti bilan birga maksimal xotira sarfi, CPU yuklanishi va energiya iste’molini ham qayd etish.

• **3.** Java tajribalarida JVM warm-up bosqichini, C++ tajribalarida kompilyator optimallashtirish parametrlarini, Python tajribalarida esa interpretator va kutubxona versiyalarini aniq ko‘rsatish.

• **4.** Natijalarni median qiymat, tarqalish ko‘rsatkichi va diagrammalar orqali taqdim etib, o‘lchovlar takrorlanuvchanligini ta’minlash.

• **5.** Kelgusida Go, Rust va JavaScript kabi tillarni ham qiyosiy tahlilga qo‘shib, tadqiqot qamrovini kengaytirish.

Mazkur yo‘nalishlarda olib boriladigan keyingi tajribalar dasturlash tillarining amaliy samaradorligi bo‘yicha aniqroq xulosalar chiqarish va turli sohalar uchun optimal texnologik tanlovni asoslash imkonini beradi.

Foydalanilgan adabiyotlar:

1. Prechelt, L. (2000). An empirical comparison of C, C++, Java, Perl, Python, Rexx. IEEE Computer, 33, 23–29. https://doi.org/10.1109/2.876288

2. Pereira, R., Couto, M., Ribeiro, F., Rua, R., Cunha, J., Fernandes, J. P., & Saraiva, J. (2017). Energy efficiency across programming languages. Science of Computer Programming, 147, 62–76. https://doi.org/10.1016/j.scico.2017.05.006

3. Python Software Foundation. Python 3 Documentation. https://docs.python.org/3/

4. ISO/IEC. (2020). ISO/IEC 14882:2020: Programming Languages — C++. International Organization for Standardization.

5. Oracle. Java Documentation. https://docs.oracle.com/en/java/

6. Oracle. The Java Virtual Machine Specification, Java SE 21 Edition. https://docs.oracle.com/javase/specs/jvms/se21/html/

7. Stroustrup, B. (2013). The C++ Programming Language (4th ed.). Addison-Wesley.

8. Lutz, M. (2013). Learning Python (5th ed.). O’Reilly Media.