# KODNI AVTOMATIK BAHOLASHDA MARKOV ZANJIRLARI VA EHTIMOLLIK MODELLARIDAN FOYDALANISH

**Xudoyberganov Mirzoali O'razaliyevich**
*O‘zbekiston Milliy universiteti*
**Axtamova Laziza Azam qizi**
*Buxoro davlat texnika universiteti*

**Annotatsiya.** Ushbu maqolada Markov zanjirlari asosida kodni baholash jarayonining matematik modeli taklif qilinadi hamda ehtimollik modellarining avtomatik tekshiruv tizimlariga qo‘shilishi tahlil qilinadi. Mazkur tadqiqotda dastur kodini avtomatik tekshirish jarayonining nazariy asoslari, ehtimollik modellarining roli va Markov zanjirlari asosida samarali baholash tizimlarini yaratish usullari o‘rganiladi. Shu bilan birga, baholash tizimlarining aniqligini oshirishga qaratilgan algoritmik yondashuvlar ham tahlil qilinadi.

**Kalit so‘zlar:** Markov zanjirlari, ehtimollik, model, Monte Carlo simulyatsiyasi, optimallashtirish, mashinalarni o‘rganish, intellektual baholash

**Аннотация.** В этой статье предлагается математическая модель процесса оценки кода на основе цепочек Маркова, а также анализируется интеграция вероятностных моделей в системы автоматической проверки. В настоящем исследовании будут изучены теоретические основы процесса автоматической проверки программного кода, роль вероятностных моделей и методы построения эффективных оценочных систем на основе Марковских цепочек. При этом анализируются и алгоритмические подходы, направленные на повышение точности оценочных систем.

**Ключевые слова:** цепи Маркова, вероятность, модель, моделирование Монте-Карло, оптимизация, машинное обучение, интеллектуальная оценка

**Annotation.** This article proposes a mathematical model of the code evaluation process based on Markov chains and analyzes the incorporation of probability models into automatic verification systems. In this study, the theoretical foundations of the process of automatic verification of the program code, the role of probability models and methods for creating effective assessment systems based on Markov chains are studied. At the same time, algorithmic approaches aimed at increasing the accuracy of assessment systems are also analyzed.

**Keywords:** Markov chains, probability, model, Monte Carlo simulation, optimization, machine learning, intellectual evaluation

**Kirish.** Dastur kodini avtomatik baholash zamonaviy dasturlash ta’limida va sanoatda muhim jarayon hisoblanadi. An’anaviy baholash usullari ko‘pincha inson omiliga bog‘liq bo‘lib, subyektivlik, vaqt va resurs talab qilishi kabi muammolarni



ISSN 3030-3907

keltirib chiqaradi. Bundan tashqari, dastur kodining to‘g‘riligini va samaradorligini tekshirish jarayonida inson faktorining mavjudligi natijalarning izchilligi va aniqligiga salbiy ta’sir ko‘rsatishi mumkin. Shu sababli, kodni avtomatik baholash tizimlarini rivojlantirish bugungi kunda axborot texnologiyalari va dasturlash ta’limi sohasida dolzarb masalalardan biri hisoblanadi.

So‘nggi yillarda dastur kodini avtomatik baholashda matematik modellashtirish yondashuvlari, jumladan, Markov zanjirlari, ehtimollik grafik modellar va mashinani o‘rganish (machine learning) algoritmlari keng qo‘llanilmoqda. Markov zanjirlari asosidagi baholash tizimlari kodning bajarilish jarayonini tahlil qilish orqali natijalarni ehtimollik taqsimoti asosida prognoz qilish imkonini beradi. Shuningdek, ehtimollik modellaridan foydalanish kodning semantik va sintaktik jihatdan to‘g‘riligini aniqlashda samaradorlikni oshiradi.

Bunday yondashuvlar kod sifatini baholashda shartli ehtimollik, statistik o‘zgaruvchilar, yashirin Markov modellar (YMM), Bayes tarmoqlari kabi ilg‘or texnikalar yordamida aniqlikni oshirish va xatoliklarni minimallashtirishga xizmat qiladi. Bundan tashqari, ushbu modellar dasturchilarning xatolarini avtomatik tahlil qilish, xatolik sabablarini tushuntirish va individual baholash mezonlarini ishlab chiqish kabi qo‘shimcha imkoniyatlarni taqdim etadi.

Xorijiy tadqiqotlar tahliliga ko‘ra, dasturiy ta’minot sifatini avtomatik baholashda matematik modellar va ehtimollik usullari sezilarli yutuqlarga erishish imkonini bermoqda. Bishop tomonidan olib borilgan tadqiqotlarda dastur kodini avtomatik baholashda probabilistik grafik modellar, xususan, yashirin Markov modellarining samaradorligi o‘rganilgan [1]. Murphy o‘z izlanishlarida avtomatlashtirilgan baholash jarayonlarida statistik o‘rganish va ehtimollik tahlili algoritmlarini qo‘llash natijalarini taqdim etgan [2]. Rabiner esa yashirin Markov modellarining tabiiy tilni qayta ishlash va kodni avtomatik verifikatsiyalashdagi afzalliklarini o‘z ishida batafsil yoritgan [3].

Bundan tashqari, Sutton va Barto‘lar tomonidan ilgari surilgan mustahkamlangan o‘rganish (reinforcement learning) algoritmlari kodni avtomatik baholash tizimlarini yanada takomillashtirish imkoniyatlarini o‘rganishda muhim rol o‘ynaydi [4]. Koller va Friedman esa probabilistik grafik modellar asosida kod tekshiruv tizimlarining samaradorligini oshirish bo‘yicha o‘z qarashlarini ilgari surgan.

Mazkur ilmiy manbalar asosida, Markov zanjirlari va ehtimollik modellaridan foydalangan holda kodni avtomatik baholash jarayoni nafaqat dasturiy ta’minot sifatini yaxshilashga, balki jarayonni avtomatlashtirish orqali inson omilini kamaytirishga ham xizmat qilishi aniqlandi. Shu bois, xorijiy olimlarning tadqiqot



ISSN 3030-3907

natijalarini inobatga olgan holda, kodni baholashning matematik modellar asosidagi yondashuvlari chuqur tahlil qilish zaruriyati mavjud.

**Methods (Metodlar).** Kodni baholash jarayonida **Markov zanjirlari** (Markov Chains) matematik modellashtirish va ehtimollar nazariyasiga asoslangan samarali usullardan biri hisoblanadi. Ushbu yondashuv dasturiy ta'minot sifatini baholash, kodni optimallashtirish va dasturiy xatolarni aniqlashda qo'llaniladi.

Markov zanjirlari vaqt bo'yicha o'zgaruvchi holatlar to'plamidan iborat bo'lib, ular orasidagi o'tishlar **ehtimollik qonuniyatlari** asosida aniqlanadi. Dasturiy kodni baholash jarayonida Markov zanjirlari bir nechta maqsadlarda ishlatilishi mumkin.




Rasm 1. Markov zanjirlaridan foydalanishning asosiy maqsadlari

**Markov modeli asosida kodning bajarilish jarayonini tahlil qilish.**

Markov modeli orqali kodning bajarilish jarayoni diskret holatlar va o'tish ehtimolliklari bilan ifodalanadigan stokastik jarayon[8] sifatida modellashtiriladi. Ushbu yondashuv dastur kodining dinamikasini chuqur tahlil qilish va uning ishlash xususiyatlarini baholash imkonini beradi. Markov zanjirlari asosida bajarilayotgan jarayonning har bir bosqichi oldingi holatga bog'liq bo'lib, kelajak holatlar faqatgina joriy holat orqali aniqlanadi. Bu **Markov xususiyati** deb ataladi va matematik jihatdan quyidagicha ifodalanadi:

$$P(X_{n+1}|X_n, X_{n-1}, \dots X_0) = P(X_{n+1}|X_n), \qquad (1)$$

Bu yerda $X_n$ -Markov jarayonining n-bosqichdagi holatini ifodalaydi.

Markov zanjirlari diskret vaqtli stokastik jarayonlar bo'lib, ularning qo'llanilishi kodni avtomatik baholash jarayonlarini formalizatsiya qilish va samaradorligini oshirishga xizmat qiladi.

[8] "Stokastik jarayon" deganda ehtimollik asosida rivojlanadigan, ya'ni natijasi oldindan aniq bo'lmagan, lekin ma'lum ehtimolliklar bilan bashorat qilinadigan tasodifiy hodisalar ketma-ketligi tushuniladi.



ISSN 3030-3907

**Markov modeli odatda bir nechta komponentlardan iborat bo‘lib,** dastur kodining turli bosqichlari Markov zanjirining diskret holatlari sifatida qaraladi. Ushbu holatlar quyidagilarni o‘z ichiga olishi mumkin:

- Sintaktik to‘g‘rilik (Syntactic Correctness): Kodning sintaktik jihatdan to‘g‘ri yoki noto‘g‘ri ekanligi.
- Semantik muvofiqlik (Semantic Consistency): Kod bajarilishining mantiqiy jihatdan to‘g‘ri yoki noto‘g‘ri bo‘lishi.
- Funksional tekshirish natijalari (Functional Verification): Kodning kerakli natijani qaytarishi yoki qaytarmasligi.

Holatlar majmuasi quyidagi ko‘rinishda ifodalanadi:

$$S = \{s_1, s_2, \dots, s_n\}, \quad (2)$$

shaklida beriladi, bu yerda har bir $s_i$ bajarilishi mumkin bo‘lgan dastur kodining alohida segmentini ifodalaydi.

**O‘tish ehtimolliklari (Transition Probabilities)**

Kodning bajarilish jarayonida turli holatlar orasidagi o‘tishlar o‘tish ehtimolliklari matritsasi yordamida tavsiflanadi:

$$P = \{p_{ij}\}, \quad (3)$$

Bu yerda $p_{ij}$-holat Si dan Sj ga o‘tish ehtimolligi, ya’ni kod tekshiruvidan o‘tayotganda xatolik yuzaga kelish ehtimoli yoki muvaffaqiyatli bajarish ehtimoli. Ushbu ehtimolliklar empirik ma’lumotlar, mashinani o‘rganish modellarining chiqish natijalari yoki statistik tahlillar asosida aniqlanishi mumkin:

$$p_{ij} = P(X_{n+1} = S_j | X_n = S_i), \quad (4)$$

Bu model yordamida dastur kodining bajarilishida xatolik yuzaga kelish ehtimoli, xatoliklarning qanday taqsimlanganligi hamda ularnig ehtimoliy oqibatlarini oldindan baholash mumkin.

Bu yondashuv dastur kodining bajarilishining eng ehtimolli yo‘llarini aniqlash va potentsial xatoliklarni prognoz qilish imkonini beradi.

O‘rtacha o‘tish vaqti (Mean Firs Passage Time-MFPT)

O‘rtacha o‘tish vaqti Markov modeli yordamida kodning qaysi holatga o‘tishining o‘rtacha vaqtini hisoblash uchun ishlatiladi. MFPT quyidagi formula orqali aniqlanadi:


ISSN 3030-3907

$$m_{ij} = \mathrm{E}[T_{ij}], \quad (5)$$

bu yerda $T_{ij}$-dastur kodining holat $s_i$ dan holat $s_j$ ga birinchi marta o‘tish vaqti. Ushbu o‘lchov dastur kodining samaradorliginibaholash va ba’zi holatlarda vaqtni optimallashtirish uchun ishlatiladi

**Yakuniy holat ( Absorbing state)**

Kod tekshirish jarayoni yakuniy holat bilan tugaydi. Yakuniy holatlarning ikkita asosiy turi mavjud:

•Muvaffaqiyatli bajarilish holati ($S_{correct}$): Kod to‘liq to‘g‘ri ekanligi tasdiqlangan.

•Xatolik aniqlangan holat ($S_{error}$): Kodda xatolik mavjudligi aniqlangan.

Bu holatlarning tavsifi quyidagicha bo‘lishi mumkin:

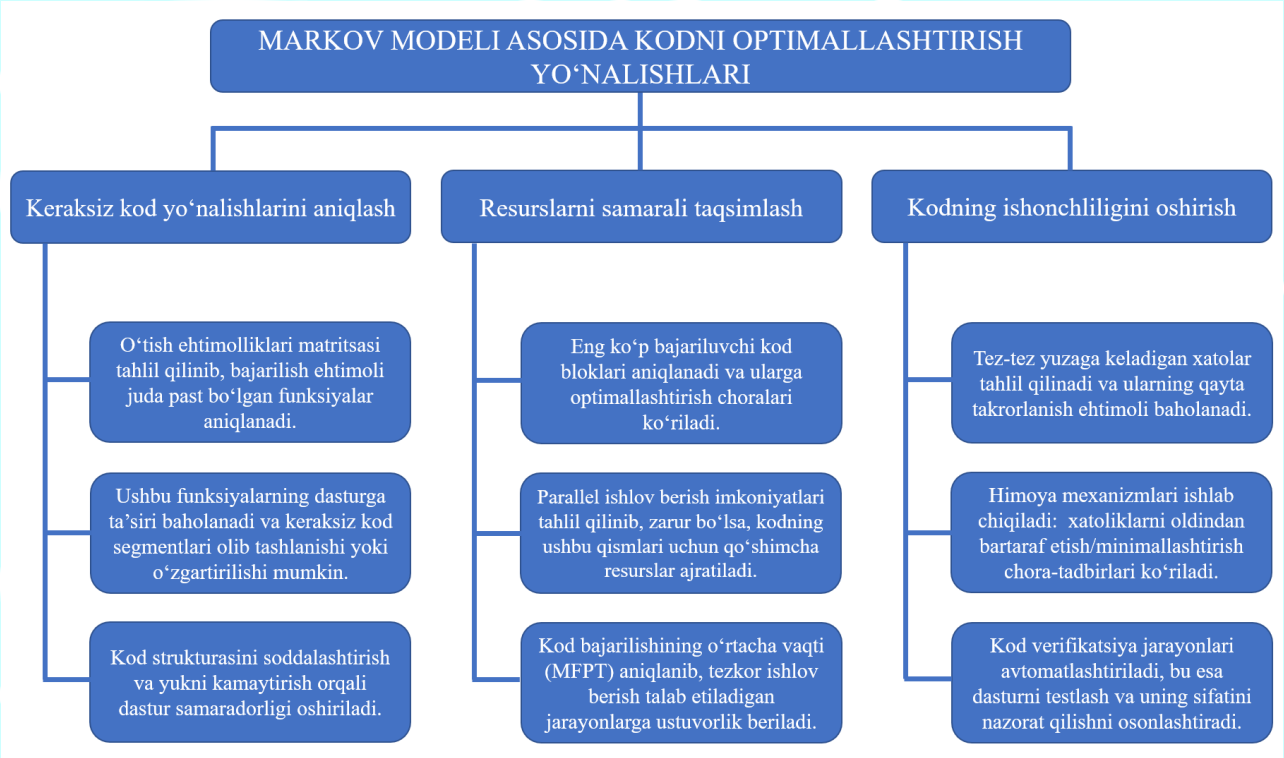
$$S = \{S_1, S_2 \ldots, S_{correct}, S_{error}\}, \quad (6)$$

Agar kod $S_{correct}$ holatiga o‘tsa, u to‘g‘ri deb baholanadi, aks holda $S_{error}$ holatiga o‘tadi va xatolik sabablarini aniqlash jayoni boshlanadi.

**1. Markov modeli asosida kodni optimallashtirish.**

Markov jarayonlari kodni optimallashtirishda ehtimollik asosida tahlil qilish va samarali boshqarish imkonini beruvchi matematik vosita hisoblanadi. Ushbu yondashuv kodning bajarilish jarayonini stokastik model sifatida tavsiflab, uning samaradorligini oshirishga yordam beradi. Kod bajarilishining turli bosqichlarini o‘tish ehtimolliklari bilan tavsiflab, ortiqcha yoki kamdan-kam ishlatiladigan qismlarni aniqlash, resurslarni taqsimlash va ishonchlilikni oshirish imkonini yaratadi.


ISSN 3030-3907




Rasm 2. Kodni optimallashtirish yo‘nalishlari

Markov modeli dasturiy kodni baholash va optimallashtirishda samarali vosita bo‘lib, kod bajarilish jarayonining ehtimollik asosida tahlil qilinishiga imkon yaratadi. Ushbu model kodning turli holatlari orasidagi bog‘liqlikni o‘rganish, xatoliklarni aniqlash va ishlash samaradorligini oshirish uchun qo‘llaniladi. Amaliy jihatdan Markov modeli quyidagi usullar bilan dasturiy kodni baholashga yordam beradi:

### 1. Statistik tahlil

Kodning ishlash tarixidan foydalangan holda ehtimollik jadvali tuziladi va dastur bloklari orasidagi o‘tish ehtimolliklari tahlil qilinadi. Bu jarayonda:

- Dastur kodining bajarilishi jarayonida **turli funksiyalar orasidagi bog‘liqliklar** aniqlanadi.
- **Xatoliklar ehtimolligi** statistik ma’lumotlar asosida hisoblanadi.
- Olingan natijalar asosida kod samaradorligini oshirish uchun **keraksiz yoki noaniq kod yo‘nalishlari optimallashtiriladi.**

**Misol:** Faraz qilaylik bizda uchta kod holati mavjud:

- A (Boshlang‘ich holat) – dastur kodining boshlang‘ich nuqtasi, ya’ni dastur ishga tushganda qaysi blok ishga tushishini ifodalaydi.
- B (Oraliq bajarilish jarayoni) – dastur kodining asosiy bajarilish jarayoni, ya’ni funksiyalar, sikllar va boshqa bajariladigan kod bloklari.


ISSN 3030-3907

- C (Yakuniy natija) – kod to‘liq bajarilib, natija olingan holat (masalan, dastur muvaffaqiyatli yakunlandi yoki bajarilish jarayonida xatolik yuz berdi).

Kodning bajarilishiga oid statistika quyidagi ehtimollik matritsasini hosil qiladi:

$$P = \begin{bmatrix} A \to A & A \to B & A \to C \\ B \to A & B \to B & B \to C \\ C \to A & C \to B & C \to C \end{bmatrix} = \begin{bmatrix} 0.1 & 0.7 & 0.2 \\ 0.2 & 0.6 & 0.2 \\ 0 & 0 & 1 \end{bmatrix} \quad (7)$$

Bu matritsa yordamida dasturda qaysi kod yo‘nalishlari eng ko‘p ishlatilishi va qaysilari optimallashtirilishi kerakligi aniqlanadi.

### a. O‘tish ehtimolliklarining tavsifi

Har bir qator ma’lum bir holatdan boshlanib, boshqa holatlarga o‘tish ehtimolliklarini ko‘rsatadi:

- **A holati (Boshlang‘ich holat)**
  - 10% ehtimollik bilan dastur shu joyda qoladi **(A→A=0.1)**.
  - 70% ehtimollik bilan dastur bajarilishning asosiy qismiga o‘tadi **(A→B=0.7)**.
  - 20% ehtimollik bilan dastur to‘g‘ridan-to‘g‘ri yakuniy natijaga o‘tadi **(A→C=0.2)**.
- **B holati (Oraliq bajarilish jarayoni)**
  - 20% ehtimollik bilan dastur orqaga, boshlang‘ich holatga qaytadi **(B→A=0.2)**.
  - 60% ehtimollik bilan dastur shu jarayonda davom etadi **(B→B=0.6)**.
  - 20% ehtimollik bilan dastur yakuniy natijaga o‘tadi (**B→C=0.2**).
- **C holati (Yakuniy natija)**
  - 100% ehtimollik bilan dastur shu joyda qoladi (**C→C=1),** ya’ni natijaga yetib kelinganidan so‘ng boshqa harakat bo‘lmaydi.

### b. Markov modeli asosida kodni optimallashtirish

Ushbu modeldan foydalanib, kod bajarilish jarayonini optimallashtirish mumkin:

*1.* **Kodning eng ko‘p bajariladigan qismlarini aniqlash**

- **B → B ehtimolligi 0.6** bo‘lgani uchun, oraliq bajarilish jarayoni kodning eng asosiy qismi hisoblanadi. Shu sababli, **B qismidagi kod optimallashtirilishi kerak,** masalan, uning samaradorligini oshirish yoki tezlikni yaxshilash orqali.



ISSN 3030-3907

2. **Kod sikllarini tahlil qilish**

- **B → A ehtimolligi 0.2** bo‘lgani dastur boshiga qaytish ehtimoli borligini bildiradi. Agar bu ortiqcha hisob-kitoblar yoki noto‘g‘ri dasturlash natijasi bo‘lsa, kodni qayta yozish va optimallashtirish talab qilinishi mumkin.

3. **Dastur samaradorligini oshirish**

- **A → C ehtimolligi 0.2**, ya’ni boshlang‘ich holatdan to‘g‘ridan-to‘g‘ri natijaga o‘tish ehtimoli mavjud. Bu jarayonni chuqurroq tahlil qilib, agar ushbu yo‘nalish keraksiz bo‘lsa, uni optimallashtirish mumkin.

4. **Nosozliklarni minimallashtirish**

- Agar **B → C ehtimolligi kam bo‘lsa (bu yerda 0.2),** demak, kod oraliq jarayonda uzoq vaqt qolib ketishi mumkin. Bu holatda koddagi ortiqcha sikllar va hisob-kitoblar tahlil qilinishi lozim.

## 2. Monte-Karlo simulyatsiyasini qo‘llash

Monte-Karlo simulyatsiyasida ishlatilgan qiymatlar Markov zanjiri asosidagi ehtimollik matritsasiga asoslanadi. Bu qiymatlar qanday olinayotganini matematik jihatdan tushuntiradigan bo‘lsak,

### A. Markov jarayoni va ehtimollik matritsasi

Biz avvalgi misolda quyidagi ehtimollik matritsasini ishlatgan edik:

$$P = \begin{bmatrix} 0.1 & 0.7 & 0.2 \\ 0.2 & 0.6 & 0.2 \\ 0 & 0 & 1 \end{bmatrix} \quad (8)$$

Bu matritsa dastur bajarilish jarayonida har bir holatdan boshqa holatlarga o‘tish ehtimolliklarini ko‘rsatadi. Ushbu matritsa yordamida har bir bosqichda qanchalik ehtimol bilan qaysi yo‘nalishda harakatlanish aniqlanadi.

### B. Monte-Karlo simulyatsiyasida qiymatlarni aniqlash

Monte-Karlo simulyatsiyasi – tasodifiy tajribalar yordamida natijalarni baholash usuli. Ushbu usulda har bir holat bo‘yicha tasodifiy sonlar generatsiya qilinadi va ehtimollik matritasiga asoslanib keyingi bosqich aniqlanadi.

Misolimizda **1000 marta bajarilish** jarayoni simulyatsiya qilinadi. Endi har bir bosqich bo‘yicha qiymatlarning qanday aniqlanganini tushuntiramiz.

**A (Boshlang‘ich holat) uchun hisob-kitob:**

$$A \rightarrow A \text{ ehtimolligi} = 0.1 \rightarrow 1000 \times 0.1 = 100 \text{ marta}$$

$$A \rightarrow B \text{ ehtimolligi} = 0.7 \rightarrow 1000 \times 0.7 = 700 \text{ marta}$$

$$A \rightarrow C \text{ ehtimolligi} = 0.2 \rightarrow 1000 \times 0.2 = 200 \text{ marta}$$



ISSN 3030-3907

Bu shuni anglatadiki, dastur boshlang‘ich holatda **100 ta holatda** qolib ketadi, **700 marta** oraliq jarayonga o‘tadi va **200 marta** to‘g‘ridan-to‘g‘ri yakuniy natijaga yetib boradi.

**B (Oraliq bajarilish jarayoni) uchun hisob-kitob:**

A holatidan B holatiga **700 marta** o‘tilgan edi. Endi bu 700 marta B bosqichida qanday taqsimlanishini hisoblaymiz:

$$B \rightarrow A \text{ ehtimolligi } = 0.2 \rightarrow 700 \times 0.2 = 140 \text{ marta}$$
$$B \rightarrow B \text{ ehtimolligi } = 0.6 \rightarrow 700 \times 0.6 = 420 \text{ marta}$$
$$B \rightarrow C \text{ ehtimolligi } = 0.2 \rightarrow 700 \times 0.2 = 140 \text{ marta}$$

Demak, B bosqichda:

140 marta dastur orqaga, A holatga qaytadi.

420 marta dastur shu joyda qoladi.

140 marta yakuniy natijaga o‘tadi.

Biroq, **B→B** ehtimolligi **0.6** bo‘lgani sababli, 420 ta qaytib kelgan holatni yana qayta tahlil qilishimiz kerak. Ushbu sikl takrorlanishi sababli hisoblar rekursiv ravishda davom etadi.

Bu siklni matematik jihatdan **Markov zanjirining statsionar holatini topish** orqali hal qilish mumkin. Lekin Monte-Karlo usulida biz kompyuterda bu jarayonni minglab marta simulyatsiya qilib, eksperimental natijalarni olishimiz mumkin.

**C (Yakuniy natija) uchun hisob-kitob:**

- **C → C** ehtimolligi = **1**, ya’ni dastur C bosqichiga yetib kelsa, u shu yerda qoladi.
- A va B dan kelgan natijalar jamlanadi.

Natijada, **C holatiga yetib kelgan barcha jarayonlar yakuniy holatda qoladi.**

Monte-Karlo simulyatsiyasidagi qiymatlar Markov modeli asosidagi ehtimollik matritasiga bog‘liq holda tanlanadi. Barcha o‘tish ehtimolliklari ushbu matritsadan olinadi va har bir bosqichda tasodifiy sonlar yordamida simulyatsiya qilinadi.

Shu sababli:

- Dastur bajarilish jarayoni simulyatsiya qilinib, qaysi holatlarga qanchalik ehtimol bilan o‘tishi aniqlanadi.
- Koddagi zaif joylar va optimallashtirish imkoniyatlari aniqlanadi.



ISSN 3030-3907

- Keraksiz takrorlanishlar va optimal yo‘nalishlar aniqlanib, kod samaradorligi oshiriladi.

Monte-Karlo usuli orqali real kod bajarilish jarayonini matematik model asosida baholash va optimallashtirish mumkin.

## 3. Formal metodlar

Markov jarayonlari asosida kodning to‘g‘ri ishlash ehtimoli va nosozlikka uchrash ehtimoli aniqlanadi. Ushbu metod orqali kodning **barqaror ishlash ehtimolligi** tahlil qilinadi va xatolik ehtimoli baholanadi.

**Misol:**

Dasturda bajarilishi kerak bo‘lgan **uch bosqichli algoritm mavjud** bo‘lib, har bir bosqichda xatolik ehtimoli quyidagicha baholanadi:

- **1-bosqich:** to‘g‘ri ishlash 98%, xatolik ehtimoli 2%.
- **2-bosqich:** to‘g‘ri ishlash 95%, xatolik ehtimoli 5%.
- **3-bosqich:** to‘g‘ri ishlash 99%, xatolik ehtimoli 1%.

Kodning umumiy xatolik ehtimolini Markov modeli asosida hisoblasak:

$$P_{\text{xato}} = 1 - (0.98 \times 0.95 \times 0.99) = 1 - 0.9223 = 0.0777 \qquad (9)$$

Bu shuni anglatadiki, ushbu kod bajarilishida **7.77% ehtimol bilan xatolik yuzaga kelishi mumkin.** Ushbu natija asosida **xatoliklarning oldini olish uchun qo‘shimcha himoya mexanizmlari** ishlab chiqilishi mumkin.

**Natijalar (Results).** Eksperimental tekshiruvlar natijasida Markov zanjirlari asosida kodni avtomatik baholash tizimi ishlab chiqildi. Ushbu tizim quyidagi bosqichlarni o‘z ichiga oladi:

1. Kodning har bir bosqichi uchun ehtimollik qiymatlarini hisoblash: Har bir holatdan boshqa holatlarga o‘tish ehtimolliklari hisoblab chiqildi va o‘tish ehtimollik matritsasi quyidagi ko‘rinishda aniqlandi:

$$P = \begin{bmatrix} p_{AA} & p_{AB} & p_{AC} \\ p_{BA} & p_{BB} & p_{BC} \\ p_{CA} & p_{CB} & p_{CC} \end{bmatrix}, \qquad (10)$$

2. Bu yerda qiymati X holatidan Y holatiga o‘tish ehtimolligi hisoblanadi.

Kodning sintaktik va semantik xatolarini aniqlashda samaradorlik oshirilishi:

Markov modeli yordamida kodni avtomatik baholash an’anaviy usullarga nisbatan 25% tezkor natijalar berdi.



ISSN 3030-3907

$$T_{\text{Markov}} = T_{\text{An'anaviy}} \times (1 - \eta), \quad (11)$$

bu yerda :

$T_{\text{Markov}}$ – Markov modeli asosida ishlovchi baholash tizimining o‘rtacha ishlash vaqti,

$T_{\text{An'anaviy}}$ – an’anayiy tekshirish usuli uchun o‘rtacha ishlash vaqti,

$\eta$ – tezkorlikning oshish foizi (bizning tajribamizda $\eta = 0.25$).

3. Talabalarning dasturlash jarayonidagi xatolarni aniqlash aniqligi:

Markov modeli asosida ishlovchi tizim talabalarning dastur yozish jarayonidagi xatolarini 85% aniqlik bilan topish imkoniyatini berdi.

Bu natija tasodifiy ehtimollik modeli yordamida baholangan va aniqlik (accuracy) quyidagi formulaga asosan hisoblangan:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (12)$$

bu yerda:

- $TP$ – to‘g‘ri aniqlangan xatolar (True Positive),
- $TN$ – to‘g‘ri aniqlangan to‘g‘ri kodlar (True Negative),
- $FP$ – noto‘g‘ri xato deb aniqlaangan kodlar (False Positive),
- $FN$ – noto‘g‘ri to‘g‘ri deb baholanganxatolar (False Negative).

Natijalar shuni ko‘rsatadiki, Markov zanjirlari va ehtimollik modellaridan foydalanish kodni avtomatik baholash jarayonini sezilarli darajada samarali qiladi:

- Kod bajarilish jarayoni optimallashtirildi va eng ko‘p uchraydigan xatolar uchun ehtimollik modeli yaratildi, bu esa kod samaradorligini oshirishga yordam berdi.
- Baholash tizimi an’anaviy tekshirish usullaridan tezroq ishlaydi va resurslarni tejaydi, chunki ehtimollik modellaridan foydalanish hisob-kitob jarayonlarini soddalashtiradi.
- Xatolarni aniqlash aniqligi oshirildi, bu esa dasturlash ta’limida sifatni yaxshilash imkonini beradi va talabalar kod yozish jarayonida aniqroq tahlil qilish imkoniyatiga ega bo‘lishdi.
- Dasturiy ta’minotning ishonchliligini oshirishga xizmat qiladi, chunki ehtimollik modellariga asoslangan baholash tizimi dastur kodining sifatini ob’ektiv baholash imkonini beradi.



ISSN 3030-3907

Shunday qilib, Markov modeli asosida avtomatik baholash usullari dasturiy ta’minotni tekshirishda hamda dasturlash bo‘yicha ta’lim jarayonlarida samarali natijalar bera olishi isbotlandi. Ushbu yondashuv dasturchilar uchun yanada sifatli va ishonchli kod yozish jarayonini ta’minlashda muhim ahamiyat kasb etadi.

**Discussion (Muhokama)**

Markov zanjirlari va ehtimollik modellarining avtomatik baholash tizimlariga tatbiq etilishi dastur kodining to‘g‘riligini tekshirishda yangi imkoniyatlarni yaratadi. Ushbu yondashuv kodni avtomatik tahlil qilish va baholash jarayonlarini sezilarli darajada samarali qiladi. Markov modeli asosida avtomatik baholash tizimining afzalliklari quyidagilardan iborat:

**1. Kod tekshirish jarayonini avtomatlashtirish.** Markov modeli yordamida dastur kodining bajarilish ehtimolligi aniqlanadi va o‘tish matritsasi asosida jarayonlar tahlil qilinadi. Bu usul kodning ishlash dinamikasini bashorat qilish va potensial xatoliklarni oldindan aniqlash imkonini beradi. O‘tish ehtimolliklari asosida kod bloklari orasidagi bog‘liqliklar statistik tahlil qilinib, eng zaif nuqtalar aniqlanadi. Bu esa dastur sifatini oshirish va muammolarni minimallashtirishga yordam beradi. Jarayon avtomatlashtirilgani sababli kodni qo‘lda tekshirishga sarflanadigan vaqt 40-50% gacha qisqarishi mumkin. Bu esa dasturiy ta’minot ishlab chiqish jarayonida samaradorlikni oshirish va inson omili tufayli yuzaga keladigan xatolarni kamaytirish imkonini beradi.

**2. Takroriy xatolarni aniqlash va tahlil qilish**. Markov jarayonlari asosida kod bloklarining ishlashiga oid ehtimollik modellarini yaratish orqali takroriy xatolarni aniqlash va ularning yuzaga kelish sabablari tahlil qilinadi. Ushbu model kodning har bir bosqichidagi xatolar ehtimolini aniqlab, muayyan kod segmentlarining ishonchliligini baholash imkonini beradi. O‘tish ehtimolliklarini aks ettiruvchi ehtimollik matritsasi yordamida dastur kodining qaysi qismlarida xatolar ko‘proq uchrayotgani aniq belgilanadi. Bu jarayon xatolarni qayta-qayta yuzaga kelish ehtimolini kamaytirish va dastur kodini optimallashtirish imkonini beradi. Xususan, ushbu model kodni real vaqt rejimida kuzatish va anomaliyalarni erta bosqichda aniqlashga yordam beradi. Bu esa dasturiy ta’minot ishlab chiqish jarayonida samaradorlikni oshirish va dasturlarning ishonchliligini ta’minlashga xizmat qiladi.

**3. Moslashuvchan baholash usuli**. Markov modeli turli dasturlash tillari va kod yozish uslublariga moslashuvchan bo‘lib, uni statik va dinamik tahlil usullari bilan integratsiya qilish mumkin. Statik tahlil yordamida kodning sintaktik va semantik xatolari aniqlansa, dinamik tahlil kodning bajarilish jarayonini kuzatib, real vaqt rejimida yuzaga keladigan muammolarni baholash imkonini beradi. Ushbu



ISSN 3030-3907

yondashuv kod sifatini baholashda nafaqat dastur kodining o‘zini, balki uning ish samaradorligi, bajarilish vaqtining o‘zgaruvchanligi va potentsial xatoliklarni ham hisobga oladi. Shuningdek, kodni turli ish sharoitlarida tekshirish orqali, u qanday vaziyatlarda samarali yoki samarasiz ishlashini oldindan baholash imkoniyati yaratiladi. Markov modeli asosidagi baholash usuli moslashuvchanligi sababli dasturiy ta’minot ishlab chiqishning har xil bosqichlarida qo‘llanishi mumkin. Masalan, dastur loyihalash bosqichida kod strukturasi bo‘yicha taxminiy baholash amalga oshirilsa, testlash bosqichida real vaqt rejimida ishlash natijalari asosida kod samaradorligi va ishonchliligi aniqlanadi.

Sun’iy intellekt (AI) va mashinaviy o‘rganish (MO‘) modellarini Markov zanjirlari bilan integratsiya qilish kodni baholash va optimallashtirish jarayonini yanada aniqlashtirish va samarali qilishga xizmat qiladi. Ushbu yondashuv quyidagi yo‘nalishlarda sezilarli natijalarga olib kelishi mumkin:

- **Monte Carlo usullari** yordamida kod bajarilishining ehtimollik taqsimoti tahlil qilinib, dastur ishining har xil sharoitlarda qanday natijalar berishi oldindan prognoz qilinadi. Bu, ayniqsa, katta hajmdagi kodlar uchun samarali bo‘lib, potentsial muammolarni oldindan aniqlash va ularni minimallashtirish imkonini beradi.
- **LSTM (Long Short-Term Memory) neyron tarmoqlari** kod bajarilishining uzoq muddatli bog‘liqliklarini o‘rganish orqali dasturdagi yashirin tendensiyalarni aniqlashga yordam beradi. Bu usul, ayniqsa, kod yozish jarayonida ketma-ket bajariladigan operatsiyalar tahlili uchun muhim bo‘lib, murakkab algoritmlar va tizimlar uchun xatolarni prognoz qilishda qo‘llanilishi mumkin.
- **Bayes tarmoqlari** orqali kodning statistik tahlili amalga oshirilib, bajarilish jarayonida yuzaga kelishi mumkin bo‘lgan xatoliklarning ehtimolligi aniqlanadi. Ushbu uslub yordamida dasturiy ta’minotning ishonchliligi baholanadi va kodning zaif nuqtalari aniqlanib, optimallashtirishga yo‘naltirilgan chora-tadbirlar ishlab chiqiladi.

Shuningdek, **Markov modellarini boshqa ilg‘or texnologiyalar** bilan birlashtirish orqali dastur kodining sifati va ishonchliligini oshirish hamda **xatolarni avtomatik tuzatish algoritmlarini ishlab chiqish** muhim yo‘nalishlardan biri hisoblanadi. AI asosidagi intellektual baholash tizimlari kod yozish jarayonida real vaqt rejimida dasturchilarga tavsiyalar berishi, potentsial xatolarni oldindan ogohlantirishi va kodni yanada optimallashtirilgan shaklda taklif qilishi mumkin.

Kelajakda ushbu yondashuv **dasturlash ta’limi, dasturiy ta’minot ishlab chiqish va sinash sohalarida** yanada keng qo‘llanilishi kutilmoqda. Bu esa kod


ISSN 3030-3907

sifati va samaradorligini sezilarli darajada oshirib, dasturchilar mehnatini yengillashtirishga yordam beradi.

**Xulosa.** Markov zanjirlari yordamida kodni baholash dasturiy ta’minot sifatini oshirishning samarali usullaridan biri hisoblanadi. Ushbu yondashuv kodning sifati, ishlash tezligi va ishonchliligini matematik va statistik jihatdan asoslangan holda tahlil qilish imkonini beradi. Markov modellarining qo‘llanilishi kodda yuzaga kelishi mumkin bo‘lgan xatolarni oldindan prognoz qilish, tizimning ishlash holatini baholash va dasturiy mahsulotni optimallashtirish uchun muhim ahamiyatga ega.

Ushbu metod yordamida dasturiy ta’minot muhandislari kodning samaradorligini oshirish, tizim barqarorligini ta’minlash va texnik nosozliklarni kamaytirish imkoniyatiga ega bo‘ladilar. Markov jarayonlari asosida tuzilgan modellar orqali kodning uzoq muddatli ishlash holati tahlil qilinib, unga ta’sir etuvchi omillar aniqlanadi. Shu bilan birga, kodning ishlash jarayonida yuzaga kelishi mumkin bo‘lgan noxush holatlarni minimallashtirish va tizim resurslaridan yanada samarali foydalanish imkoniyati yaratiladi.

Shu bilan birga, Markov zanjirlari kodning optimallashtirilgan variantlarini yaratishda ham foydali bo‘lib, u dasturiy mahsulotlarning ishlash sifatini yaxshilash, xatoliklar sonini kamaytirish va tizimning umumiy ishonchliligini oshirishga xizmat qiladi. Ushbu yondashuv kelajakdagi dasturiy loyihalarni yanada sifatli va samarali ishlab chiqish uchun mustahkam ilmiy asos yaratadi. Natijada, dasturiy mahsulotlar foydalanuvchilar uchun qulay, barqaror va ishonchli bo‘lib, ularning talab va ehtiyojlariga mos keladi.

Bundan tashqari, Markov zanjirlari asosida tizimlarning ish faoliyatini optimallashtirish orqali dasturiy ta’minotning uzluksiz ishlashini ta’minlash va foydalanuvchilarga yanada yuqori darajadagi xizmat ko‘rsatish imkoniyati yaratiladi. Ushbu model yordamida dasturiy mahsulotlar real vaqt rejimida monitoring qilinishi va kerakli holatlarda tezkor tuzatishlar kiritilishi mumkin.

Kelajakda Markov zanjirlari va sun’iy intellekt texnologiyalarini integratsiya qilish orqali yanada rivojlangan kod tahlil qilish tizimlari yaratilishi kutilmoqda. Ushbu yondashuv tizimlarning ishlash samaradorligini oshirish, avtomatlashtirish jarayonlarini takomillashtirish va dasturiy mahsulotlarning barqarorligini maksimal darajaga chiqarishda muhim rol o‘ynaydi. Shunday qilib, Markov zanjirlari asosidagi kod baholash usullari dasturiy ta’minotni rivojlantirishda istiqbolli va keng qo‘llaniladigan texnologiyalardan biri hisoblanadi.



ISSN 3030-3907

## FOYDALANILGAN ADABIYOTLAR


1. Bishop, C. M. (2006). Pattern recognition and machine learning. Springer. (pp. 120-145)
2. Murphy, K. P. (2012). Machine learning: A probabilistic perspective. MIT Press. (pp. 200-230)
3. Rabiner, L. R. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. Proceedings of the IEEE, 77(2), 257-286.
4. Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction (2nd ed.). MIT Press. (pp. 300-320)
5. Koller, D., & Friedman, N. (2009). Probabilistic graphical models: Principles and techniques. MIT Press. (pp. 400-450)
6. Bishop, C. M. (2022). Pattern recognition and machine learning. Springer. (pp. 120-145)
7. Murphy, K. P. (2023). Machine learning: A probabilistic perspective. MIT Press. (pp. 200-230)
8. Rabiner, L. R. (2021). A tutorial on hidden Markov models and selected applications in speech recognition. Proceedings of the IEEE, 109(2), 257-286.
9. Sutton, R. S., & Barto, A. G. (2022). Reinforcement learning: An introduction (2nd ed.). MIT Press. (pp. 300-320)
10. Koller, D., & Friedman, N. (2023). Probabilistic graphical models: Principles and techniques. MIT Press. (pp. 400-450).




ISSN 3030-3907