AMALIY TOPSHIRIQLARNI AVTOMATIK TEKSHIRISHDA XATONI LOKALIZATSIYA QILISH ALGORITMLARI

[1]Xudoyberganov M.O‘., [2]Axtamova L.A.

[1]O‘zbekiston Milliy universiteti

[2]Buxoro davlat texnika universiteti

Annotatsiya. *Ushbu tezis amaliy dasturlash topshiriqlarini avtomatik tekshirish jarayonida xatoni* ***lokalizatsiya qilish*** *uchun gibrid yondashuvni taklif etadi. Tizim spektrga asoslangan lokalizatsiya (SBFL), dinamik slicing, selektiv mutatsiya-ga asoslangan lokalizatsiya (MBFL) va invariant/ketma-ketlik tahlillarini bir pipeline’da birlashtiradi hamda natijani vaznli agregatsiya yoki learning-to-rank orqali reytinglaydi. Yondashuv diagnostik izohlarni boyitib, “tekshirish–feedback–qayta topshirish” siklini qisqartiradi va real kurs sharoitiga mos, ko‘chiriladigan yechim taqdim etadi. Kelgusida coverage-guided fuzzing va rubrikaga bog‘langan izohlar bilan izohlanishni yanada kuchaytirish rejalashtirilgan.*

Kalit so‘zlar: *fault localization; spectrum-based FL; dynamic slicing; mutation-based FL; invariant; EXAM score; learning-to-rank; autograder*

Dasturlashga oid amaliy topshiriqlarni avtomatik tekshirish jarayonida “Muvaffaqiyatsiz” degan signalning o‘zi yetarli emas — talaba yoki o‘qituvchiga **xatoni aniq joyi** (qatordan tortib, ifoda/predikat darajasigacha) va **sababiy sababi** ko‘rsatilsa, tuzatish sikli sezilarli qisqaradi. Ushbu tezis avtomatik tekshiruv tizimlarida **xatoni lokalizatsiya qilish (fault localization, FL)** uchun gibrid yondashuvni taklif etadi: **spektrga asoslangan FL (SBFL)**, **dinamik slicing**, **mutatsiya-ga asoslangan FL (MBFL)**, hamda **invariant/ketma-ketlik tahlili** signallarini bir pipeline’da birlashtirib, ularni vaznli agregatsiya yoki o‘rganishga asoslangan reyting bilan konsolidatsiya qiladi. Maqsad — Top-k aniqlikni, EXAM

skori (qanchalik kam kod ko‘rilib xato topilishi) va diagnostik tushuntirish sifatini oshirishdan iborat.

1) Spektrga asoslangan lokalizatsiya (SBFL). SBFL’ning g‘oyasi shundan iboratki, qaysi testlar yiqilgan (*fail*) va qaysi kod bayonotlari ushbu testlar davomida bajarilgan bo‘lsa, shu bayonotlar “xatoga yaqinroq” deb hisoblanadi; aksincha, o‘tgan (*pass*) testlar tez-tez bajaradigan bayonotlar “kamroq shubhali” bo‘ladi. Buning uchun avval qamrov matritsasi tuziladi: har bir test ttt va bayonot sss uchun $C_{t,s}$ $\{0,1\}$ (bajarilgan/bajarilmagan). Har bir bayonot bo‘yicha quyidagi sanoqlar olinadi:

- n_{ef}: failing testlar ichida **s** bajarilganlari soni;
- n_{ep}: passing testlar ichida **s** bajarilganlari soni;
- n_{ef}: failing testlar ichida **s** bajarilmaganlari soni;
- n_{ep}: passing testlar ichida **s** bajarilmaganlari soni.

So‘ng har bir sss uchun shubhalilik darajasi hisoblanadi. Masalan, ko‘p qo‘llaniladigan **Ochiai** ko‘rsatkichi:

$$\mathrm{SUS}_{Ochiai}(S) = \frac{n_{ef}}{\sqrt{(n_{ef} + n_{nf})(n_{ef} + n_{ep})}},$$

ya’ni **s** bayonotini ko‘p yiqilgan testlar bajarsa (n_{ef} ↑) va uni ko‘p o‘tgan testlar bajarmasa (n_{ef} ↓), shubhalilik ortadi. Muqobil formulalar ham bor: Tarantula

$$\mathrm{SUS}_{\mathrm{Tar}}(S) = \frac{n_{ef}\,/(n_{ef} + n_{nf})}{n_{ef}\,/(n_{ef} + n_{nf}) + n_{ep}\,/(n_{ep} + n_{np})},$$

Amalda bir nechta formulani sinab ko‘rib, Top-k reytinglar bo‘yicha bir xil xulq ko‘rsatadiganini tanlash foydali.

2) Dinamik slicing (PDG/SDG). Xatoga olib kelgan konkret test(lar) uchun dastur qaramlik grafigi (Program Dependence Graph) bo‘yicha **slicing criterion**

⟨nuqta,o‘zgaruvchi⟩\langle \text{nuqta}, \text{o‘zgaruvchi}\rangle⟨nuqta,o‘zgaruvchi⟩ tanlanadi va sababiy izlar orqaga yurib (backward slice) qisqartiriladi. Bu SBFL qaytargan yuqori reytingli satrlar atrofida “zararli kontekst”ni ajratib, diagnostika maydonini keskin toraytiradi.

3) Mutatsiya-ga asoslangan lokalizatsiya (MBFL). Talaba kodi atrofida yengil **mutatsiyalar** (shartni teskarilash, operator almashtirish, if-remove…) qo‘llanadi. Agar xatoga olib kelgan test muayyan joyga qo‘yilgan mutatsiyani “davolasa” (ya’ni endi test o‘tib ketsa), ushbu joyning shubhaliligi ortadi. Biz **Metallaxis** tipidagi ko‘rsatkichdan foydalanamiz: “mutant-kill” signali qanchalik ko‘p bo‘lsa, satr shunchalik shubhali.

4) Invariant va ketma-ketlik tahlili. Property-based testlar va **invariant kuzatuv** (masalan, qiymat diapazoni, null-bo‘lmaslik, monotoniya) bilan buzilayotgan predikatlar, shuningdek **Prefix-span/Markov** tahlili bilan xatoga eltuvchi odatiy trayektoriyalar aniqlanadi. Bu signallar “qaysi shart buzildi?” degan diagnostik savolga to‘g‘ridan-to‘g‘ri javob beradi.

5) Reytingni birlashtirish va budjetlash. Har komponentdan olingan sus(s) normalizatsiya qilinib, **vaznli agregatsiya** yoki **learning-to-rank** (LambdaMART kabi) bilan yakuniy reyting beriladi. Hisoblash xarajatini boshqarish uchun **budjetli MBFL** (faqat Top-N SBFL satrlarda mutatsiya), **selektiv slicing** va parallel konteynerlar qo‘llanadi. Granulyarlik satr → predikat → ifoda darajalarida moslashadi.

Jadval 1.

Eksperimental sozlama

Bo‘lim	Mazmuni
Ma’lumotlar	3 ta kirish darajasidagi topshiriq (Python/Java), **180 talaba**, jami **~540** yakuniy topshiriq; har biri uchun **>20** avtotes t (chekka holatlar bilan)
Bazis	**SBFL–Ochiai (yolg‘iz)** + oddiy **stack-trace**
Ko‘rsatkichlar	**Top-k aniqlik** (Top-1/Top-3), **MRR**, **EXAM** median (% ko‘rilgan kod), **qo‘shimcha vaqt** (bazisga nisbatan), **diagnostik**

	izoh sifatining ekspert balli (5-lik)
Muhit	Linux konteynerlar, **2 vCPU / 2 GB RAM**, **30–60 s** time-limit; MBFL uchun **selektiv operatorlar**

Jadval 1.

Dastlabki (bazis → gibrid) natijalar

Ko‘rsatkich	Bazis	Gibrid	Δ (farq)
Top-1 aniqlik	54%	**68%**	**+14 p.p.**
Top-3 aniqlik	76%	**88%**	**+12 p.p.**
EXAM (median, % kod ko‘rildi)	6.1%	**3.4%**	**−2.7 p.p. (≈−44%)**
MRR	0.69	**0.81**	**+0.12**
Qo‘shimcha vaqt (overhead)	0%	**+19%** *(selektiv strategiya bilan <+10%)*	—
Diagnostik izoh balli (1–5)	3.2	**4.1**	**+0.9**

Natijalar SBFL+MBFL+slicing kombinatsiyasining sinergiyasini ko‘rsatadi: SBFL **tez va keng ko‘lamli** filtrlash beradi, slicing **sababiy kontekst**ni aniq ajratadi, MBFL esa “agar shart o‘zgarsa, test o‘tadimi?” savoli bilan **kontrafaktual** dalil keltiradi. Invariant va ketma-ketlik tahlillari diagnostikani izohlanadigan qiladi. Cheklovlar: (i) MBFL hisoblash jihatdan qimmat — faqat Top-N satrda ishlatish lozim; (ii) qamrov past bo‘lsa SBFL sifati tushadi — property-based testlar bilan to‘ldirish kerak; (iii) slicing natijasi til/tool-ga sezgir; (iv) flaky testlar reytingni ifloslantiradi — testlarning barqarorligi monitoringi zarur. Adolat nuqtayi nazardan, lokalizatsiya **talabani jazolash** emas, balki **yo‘l-yo‘riq** berish vositasi sifatida ishlatiladi: tizim tavsiya qiladi, yakuniy tasdiq o‘qituvchi/assistentga qoladi.

Ushbu ishda taklif etilgan gibrid yondashuv — SBFL (spektrga asoslangan lokalizatsiya), dinamik slicing, selektiv MBFL (mutatsiya-ga asoslangan lokalizatsiya) va invariant/ketma-ketlik tahlillari — avtomatik tekshiruv tizimida xatoni topish jarayonini sezilarli darajada samaralashtirdi. Pilot natijalarda Top-

1/Top-3 aniqlikning oshishi, MRR ko‘rsatkichining yaxshilanishi va EXAM medianining kamayishi diagnostika maydonini toraytirib, “qayerdan boshlash kerak?” savoliga aniqroq javob beradi. Shuningdek, izohlanadigan (explainable) feedback boyidi: invariant buzilishlari va slicing asosidagi kontekst talabaning tuzatish qarorini tezlashtirdi va “tekshirish–feedback–qayta topshirish” siklini qisqartirdi.

Hisoblash budjeti selektiv strategiyalar bilan boshqarildi: MBFL faqat SBFL qaytargan Top-N kandidatlarda qo‘llanib, parallel konteynerlar yordamida umumiy kechikish amaliy jihatdan maqbul darajada ushlab turildi. Bu yondashuv real kurs sharoitiga mos keladi: mavjud autograder infratuzilmasiga minimal o‘zgartirishlar bilan integratsiya qilinadi, LMS/Git/CI loglari asosida replikatsiya va audit izlari ta’minlanadi.

Shu bilan birga, cheklovlar ham mavjud: test qamrovi yetarli bo‘lmaganda SBFL signali sustlashishi, flaky testlar reytingni ifloslantirishi, slicing natijasining til/strumentga sezgirligi va ko‘p-xatoli holatlarda signallarning aralashishi. Bu risklar property-based testlar bilan qamrovni kengaytirish, test barqarorligini monitoring qilish, Top-N fokuslash va klasterlash orqali bosqichma-bosqich yumshatiladi. Adolat va akademik halollik nuqtayi nazaridan, tizim tavsiya beruvchi rolni bajaradi, yakuniy tasdiq o‘qituvchi/assistentga tegishli bo‘lib qoladi (human-in-the-loop).

Kelgusidagi ishlar yo‘nalishi uch qatlamda belgilandi: (i) **sinov mexanizmlari** — coverage-guided fuzzing va differential testing orqali chekka holatlarni ko‘proq qamrab olish; (ii) **reytingni o‘rganish** — learning-to-rank modellari natijalarini kurslararo transfer qilish, Pareto muvozanatini (aniqlik–vaqt–adolat) A/B sinovlar bilan kalibrlash; (iii) **izohlanish va pedagogik moslash** — rubrikaga bog‘langan, nuqtali (granular) tushuntirishlarni avtomatlashtirish va erta-xavf flaglash (early-risk) modellari bilan ko‘maklashuvchi dashboard’ni boyitish. Umuman olganda, gibrid lokalizatsiya yondashuvi baholashning aniqligi va izohlanishini oshirib, amaliy ta’limda avtomatik tekshiruvni yanada ishonchli, adolatli va ko‘chiriladigan qiladi.

ADABIYOTLAR

1. *Xudoyberganov MO & Axtamova LA (2025). KODNI AVTOMATIK BAHOLASHDA MARKOV ZANJIRLARI VA EHTIMOLLIK MODELLARIDAN FOYDALANISH. Fanni rivojlantirish, 6(6), 406-420-betlar. https://doi.org/0*

2. Nafasov, M., Axtamova, L., & Sadullayeva, F. (2023, May). TA'LIM KONTENTINI TAQDIM ETISH UCHUN MOBIL ILOVALAR YARATISHNING ASOSIY TAMOYILLARI. In *International Scientific and Practical Conference on Algorithms and Current Problems of Programming*.

3. *Nafasov MM & Axtamova. A. (2024). DASTURLASH VAZIFALARINI AVTOMATLI TEKSHIRISH. Fan taraqqiyoti, 6(2), 95-103-betlar. https://doi.org/0*

4. *Jones, J. A., Harrold, M. J., & Stasko, J. (2002, May). Visualization of test information to assist fault localization.* Proceedings of the 24th International Conference on Software Engineering (ICSE) *(pp. 467–477). ACM. https://doi.org/10.1145/581339.581397*

5. *Abreu, R., Zoeteweij, P., & van Gemund, A. J. C. (2006, September). An evaluation of similarity coefficients for spectrum-based fault localization.* Proceedings of the 12th Pacific Rim International Symposium on Dependable Computing (PRDC'06) *(pp. 39–46). IEEE. https://doi.org/10.1109/PRDC.2006.18*

6. *Papadakis, M., Le Traon, Y., & Harman, M. (2015, May). Metallaxis-FL: Mutation-based fault localization.* Proceedings of the 2015 International Symposium on Software Testing and Analysis (ISSTA) *(pp. 449–460). ACM. https://doi.org/10.1145/2771783.2771809*

7. *Weiser, M. (1981, July). Program slicing.* Proceedings of the 5th International Conference on Software Engineering (ICSE) *(pp. 439–449). IEEE. https://doi.org/10.1109/ICSE.1981.756102*

8. *Ball, T., & Larus, J. R. (1996, May). Efficient path profiling.* Proceedings of the 29th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO-29) *(pp. 46–57). IEEE. https://doi.org/10.1109/MICRO.1996.566456*

9. *Zeller, A., & Hildebrandt, R. (2002, May). Simplifying and isolating failure-inducing input.* IEEE Transactions on Software Engineering, 28*(2), 183–200. https://doi.org/10.1109/32.988498*