Bírálat

Beszédes Árpád

Code Analysis Techniques for Debugging and Software Maintenance

című MTA doktori dolgozatáról

1. A témaválasztás időszerűsége

A dolgozat egy aktuális és releváns kutatási irányzat problémáival foglalkozik, amely szorosan kapcsolódik az informatikai rendszerek minőségbiztosításához. A választott téma nemcsak tudományos, hanem gyakorlati szempontból is fontos, a dolgozatban vizsgált kérdések közvetlenül is érintik az ipari alkalmazásokat és ezen keresztül a társadalmi igényeket. A szerző bemutatja, hogy a vizsgált problémák milyen nemzetközi és hazai összefüggésekben értelmezhetőek. Ez a kontextus megteremti a dolgozat indokoltságát, és alátámasztja az elvégzett kutatások szükségességét.

2. Szakirodalom, módszertan

A munkásságot bemutató dolgozat 7 altézis köré épül három téziscsoportban, az őket alkotó kiemelt közlemények száma 11 (a szerző 13-at sorol fel dupla redundanciával), összesen 97 független hivatkozással (MTMT, 2025. szeptember 3):

- Tézis 1.1: **Graph-Less Dynamic Program Slicing**

  Árpád Beszédes, Tamás Gergely, and Tibor Gyimóthy. Graph-less dynamic dependence-based dynamic slicing algorithms. In *Proceedings of the Sixth IEEE International Workshop on Source Code Analysis and Manipulation (SCAM'06)*, pages 21–30, September 2006.
  Független hivatkozások száma: 18

- Tézis 1.2: **Dynamic Function Coupling**

  Árpád Beszédes, Tamás Gergely, Szabolcs Faragó, Tibor Gyimóthy, and Ferenc Fischer. The Dynamic Function Coupling metric and its use in software evolution. In Proceedings of the 11th European Conference on Software Maintenance and Reengineering (CSMR'07), pages 103–112, March 2007.
  Független hivatkozások száma: 34

- Tézis 2: **Computation and Analysis of Dependence Clusters**

Árpád Beszédes, Lajos Schrettner, Béla Csaba, Tamás Gergely, Judit Jász, and Tibor Gyimóthy. Empirical investigation of SEA-based dependence cluster properties. In *Proceedings of the 13th IEEE International Working Conference on Source Code Analysis and Manipulation (SCAM'13)*, pages 1–10, September 2013.
Független hivatkozások száma: 1

Árpád Beszédes, Lajos Schrettner, Béla Csaba, Tamás Gergely, Judit Jász, and Tibor Gyimóthy. Empirical investigation of SEA-based dependence cluster properties. *Science of Computer Programming*, 105(0):3 – 25, 2015. Special Issue on SCAM'13.

David Binkley, Árpád Beszédes, Syed Islam, Judit Jász, and Béla Vancsics. Uncovering dependence clusters and linchpin functions. In *Proceedings of the 31th IEEE International Conference on Software Maintenance and Evolution (ICSME'15)*, pages 141–150, September 2015.
Független hivatkozások száma: 1

Lajos Schrettner, Judit Jász, Tamás Gergely, Árpád Beszédes, and Tibor Gyimóthy. Impact analysis in the presence of dependence clusters using Static Execute After in WebKit. In *Proceedings of the 12th IEEE International Working Conference on Source Code Analysis and Manipulation (SCAM'12)*, pages 24–33, September 2012.
Független hivatkozások száma: 9

Lajos Schrettner, Judit Jász, Tamás Gergely, Árpád Beszédes, and Tibor Gyimóthy. Impact analysis in the presence of dependence clusters using Static Execute After in WebKit. *Journal of Software: Evolution and Process*, 26(6):569–588, June 2014. Special Issue on SCAM'12.

- Tézis 3.1: **Reliable Code Coverage Measurement**

Ferenc Horváth, Tamás Gergely, Árpád Beszédes, Dávid Tengeri, Gergő Balogh, and Tibor Gyimóthy. Code coverage differences of Java bytecode and source code instrumentation tools. *Software Quality Journal*, 27(1):79–123, 2019.
Független hivatkozások száma: 18

Dávid Tengeri, Ferenc Horváth, Árpád Beszédes, Tamás Gergely, and Tibor Gyimóthy. Negative effects of bytecode instrumentation on Java source code coverage. In *Proceedings of the IEEE 23rd International Conference on Software Analysis, Evolution, and Reengineering (SANER'16)*, pages 225–235, March 2016.
Független hivatkozások száma: 10

- Tézis 3.2: **Use of Call Chains in Spectrum-Based Fault Localization**

Árpád Beszédes, Ferenc Horváth, Massimiliano Di Penta, and Tibor Gyimóthy. Leveraging contextual information from function call chains to improve fault

localization. In *Proceedings of the 27th IEEE International Conference on Software Analysis, Evolution, and Reengineering (SANER'20)*, pages 468–479, February 2020.
Független hivatkozások száma: 6

- Tézis 3.3: **Use of Call Frequencies in Spectrum-Based Fault Localization**

  Béla Vancsics, Ferenc Horváth, Attila Szatmári, and Árpád Beszédes. Call frequency-based fault localization. In *Proceedings of the 28th IEEE International Conference on Software Analysis, Evolution, and Reengineering (SANER'21)*, pages 365–376, March 2021.
  Független hivatkozások száma: 5

  Béla Vancsics, Ferenc Horváth, Attila Szatmári, and Árpád Beszédes. Fault localization using function call frequencies. *The Journal of Systems and Software*, 193:111429, 2022.
  Független hivatkozások száma: 4

- Tézis 7 (T3.4): **Use of Dynamic Slicing in Spectrum-Based Fault Localization**

  Péter Attila Soha, Tamás Gergely, Ferenc Horváth, Béla Vancsics, and Árpád Beszédes. A case against coverage-based program spectra. In *Proceedings of the 16th IEEE International Conference on Software Testing, Verification and Validation (ICST'23)*, pages 13–24, April 2023.
  Független hivatkozások száma: 1

3. A tézisek értékelése

A bevezető második sorában lévő állítás, miszerint a különféle debug technikák a szoftverfejlesztési költségek 50-70%-át teszik ki, erősen túlzó. Sok tényezőtől függően a realitás inkább 30-50%. Modern tervezési technikák és menedzselési módszertanok használata esetén ez még kevesebb is lehet.

Kérdés: A dinamikus szeletek kiszámítása során mekkora előnnyel jár a függőségi gráf helyett egyéb adatstruktúrák használata? Vannak-e erről valós ipari méretű statisztikák?

A bemutatott dinamikus függvénykapcsolási metrika értékes a szoftverrendszerek futásidejű viselkedésének megismeréséhez, különösen a karbantarthatóság javítása és az összetett dinamikus kölcsönhatások megértése szempontjából. Alkalmazása elsősorban akadémiai és kutatási kontextusban történik, ipari környezetben nem terjedt el széles körben. Ez elsősorban a számítási bonyolultsága és a szükséges futásidejű adatgyűjtés miatt van így.

Kérdés: Vajon a különböző architektúra típusok esetén mit mutat a DFC metrika?

A negyedik fejezet elején ismételten bevezetésre kerül a programszeletelés fogalomrendszere. A szoftverarchitektúrák strukturális vizsgálata kutatásának kezdete a 2000-es évek elejére tehető, ahol a kód elemei közötti statikus függőségek (pl. függvényhívások, változóhasználat, modulkapcsolatok) gráf reprezentációit vizsgálták. Az így alkotott függőségi gráfok sok szempontból elemezhetők, például a szoftvermodulok automatikus klaszterezése segít a nagy rendszerek átláthatóságában. Segítségével feltárhatók logikai alrendszerek, erős moduláris kapcsolatok és rejtett architekturális egységek. Ezt a kutatási irányt követi, folytatja a második téziscsoport. Érdekes eredménynek tartom a függőségi klaszterek szeletekre vonatkozó vizsgálati eredményeit.

Kérdés: Vannak-e mérések arról, hogy milyen mértékben támogatja a klaszterizációs metrika a karbantartók és fejlesztők döntéseit, például refaktorálás vagy újratervezés során?

A „linchpin pontok” alatt általában olyan kulcscsomópontokat értünk egy szoftver függőségi gráfban, amelyek eltávolítása vagy meghibásodása aránytalanul nagy hatással van a rendszer egészére (pl. sok modul rajtuk keresztül érhető el, vagy erős összekötő szerepük van). Az olyan gráfelméleti módszerek, mint a „betweenness centrality”, „degree centrality” vagy „articulation points” (vágópontok) hatékonyan azonosítják a kulcspontokat, utóbbi lineáris időben fut, és pontosan azokat a csomópontokat keresi, amelyek eltávolítása több komponensre bontja a gráfot. Nagy, sűrű függőségi gráfokban a Laplace-mátrix és spektrális centralitás elemzésével azonosíthatók azok a pontok, amelyek a gráf „kohézióját” tartják fenn.

Kérdés: A dolgozatban bemutatott heurisztikus megoldás hogyan viszonyul a fent felsorolt módszerekhez?

Kérdés: A statikus gráf alapján nem mindig derül ki a valós futásidejű centralitás, ezért a statikus és dinamikus metrikák kombinációja javasolt. Vannak-e ezirányú kísérleti eredmények, tervez-e ilyet?

Kifejezetten örültem a hatáselemzéssel foglalkozó fejezetnek, több ilyen kísérletre lenne szükség az ipar számára. Az eredmények értékesek, bár a szakemberek számára talán nem meglepőek.

Java projektek esetén, a forráskód és a bytekód instrumentálása összehasonlítása kifejezetten fontos a teszteredmények validálhatósága szempontjából. A mérésekre szolgáló eszközök (JaCoCo, Clover) különbözőségeinek felderítése közelebb viheti a kutatói/fejlesztői közösséget a hatékonyabb és pontosabb mérések elvégzése felé. Ezt a fejezetet is értékesnek tartom.

A hibafelderítés hatékonyságának növelése érdekében a 8. fejezet a függvény szintű hívási láncokat vizsgálja. Ha a hívási láncokat leíró modellnek a véges automata alapú modelleket választjuk, egy már meglévő és alaposan kutatott területhez érünk. Igaz, itt skálázódási problémák léphetnek fel. Például a 8.2 feladatban lévő hibát LLM-mel és action-state teszteszközzel kb. másfél perc megtalálni.

A dolgozat utolsó fejezetében bemutatott kutatás megerősíti, hogy a találat-alapú spektrumok egyszerű kiterjesztése végrehajtási statisztikákkal nem javítja a hibakeresés hatékonyságát. Ehelyett a szerzők az Egyedi Legmélyebb Hívási Verem koncepcióját vezetik

be, amely a tesztvégrehajtás során rögzíti a veremállapotokat és számolja a metódushívásokat, így csökkentve a ciklusok miatt keletkező ismétlések zaját. Az empirikus eredmények azt mutatják, hogy ez a megközelítés javítja a találat-alapú módszereket. A spektrummetrikák és kockázati formulák ilyen adaptálása újszerű hozzájárulás a spektrum-alapú hibakeresés területén, és új kutatási irányokat nyit meg, például a módszer finomabb szintekre (utasítások) való kiterjesztését vagy további kontextuális információk bevonását.

Kérdés: Vannak-e ismeretek arról, hogy a javasolt hibalokalizációs technikáknak az adat és vezérlés bonyolultságot tekintve milyenek a skálázódási tulajdonságai?

4. Általános értékelés

A szakirodalmi áttekintés széles körű, a szerző sok releváns forrást dolgoz fel, és azokból következetesen építkezik. Kiemelendő, hogy nemcsak klasszikus, hanem modern, nemzetközi publikációkat is beemelt az elemzésekbe, így a dolgozat jól tükrözi a kutatási téma aktuális állását. Ennek ellenére az áttekintésben több esetben inkább a leíró jelleg dominál, a kritikai összehasonlítás és az eltérő nézőpontok mélyebb elemzése háttérbe szorul. Így az olvasó néha nehezebben tudja megítélni, hogy a szerző pontosan mely meglévő eredményekhez kapcsolódik szervesen, és hol kíván új tudományos hozzájárulást adni.

A dolgozat módszertani része logikus felépítésű, a választott kutatási technikák megfelelőek. A szerző bemutatja az alkalmazott eszközöket, kísérleti beállításokat, illetve az elemzési lépéseket. Pozitívum, hogy a vizsgálati folyamatok gyakran többféle megközelítést is tartalmaznak, ami növeli az eredmények érvényességét. Kritikai észrevételként ugyanakkor megjegyezhető, hogy a szerző helyenként túl technikai részletezésbe bocsátkozik, míg más, a megértést segítő elemek (pl. ábrák, folyamatábrák) hiányoznak, vagy szűkszavúak. A dolgozatban bizonyos helyeken a célkitűzések túl általánosan jelennek meg, ami megnehezíti annak egyértelmű követését, hogy a szerző pontosan milyen kutatási hipotéziseket kíván igazolni vagy cáfolni. A szerző sokszor implicit módon kezeli azokat az előfeltevéseket, amelyek az érvényesség szempontjából lényegesek és nem kellőképp alátámasztottak (pl. adatforrás megbízhatósága, szimuláció korlátai, mérési hibák kezelése). Ezen túlmenően, a szerző által választott mérőszámok és kiértékelési módszerek nem minden esetben indokoltak részletesen, ami gyengítheti az eredmények általánosíthatóságát. A következtetések megfogalmazásánál néhol túlzott az általánosítás: nem minden esetben látszik, hogy a kapott eredmények mennyire robosztusak eltérő körülmények között. Ez különösen fontos lenne, mivel a gyakorlatban a szoftverrendszerek általában változó környezetekben működnek.

A dolgozat erőssége, hogy az elméleti rész és a gyakorlati alkalmazás jól kiegészíti egymást, így a kutatás nem csupán absztrakt szinten marad, hanem valódi ipari és társadalmi relevanciával bír. A dolgozat további erénye, hogy egy olyan módszertani keretet mutat be, amelyben a szerző a kvalitatív és kvantitatív megközelítéseket kombinálja. A külföldi kutatásokkal való párhuzamok bizonyítják, hogy a dolgozat következtetései szélesebb kontextusban is megállják a helyüket.

A dolgozat tudományosan megalapozott kritikát ad a programszeletelés jelenlegi állapotáról. Tematikus felépítése, aprólékos hivatkozásai és az elmélet és a gyakorlat közötti szakadék

világos megfogalmazása dicséretre méltó. A központi állítás – miszerint a szeletelés mai fontosságát különböző technikai és gyakorlati kihívások mérséklik – jól megalapozott és meggyőző.

A dokumentum hatása tovább növelhető lenne, ha tágítaná a hatókörét, beépítve a szeletelés sikereinek kiegyensúlyozottabb nézetét, a lehetséges jövőbeni megoldások megvitatását, valamint konkrétabb, szemléltető példákat. Egy átfogóbb tudományos érvelés kitérhetne azokra a területekre is, ahol a szeletelés vitathatatlanul sikeres volt. Például a szeletelés szerepének megvitatása a biztonsági elemzésben (pl. sebezhetőség-észlelés) vagy olyan specifikus területeken, mint a beágyazott rendszerek, ahol a pontos vezérlési folyamat kezelhetőbb, értékes kontextust és árnyalatokat adna. A dolgozat hatékonyan diagnosztizálja a problémákat, de kevésbé írja elő a jövőbeni kutatási irányokat. Például, hogyan segíthetnek a gépi tanulás vagy a statikus elemzés terén elért fejlődések a skálázhatósági és precizitási kihívások leküzdésében a nagy, dinamikus kódbázisokon? Hogyan lehetne egy szeleteléshez hasonló technikát integrálni a modern folyamatos integrációs (CI) folyamatokba, hogy a fejlesztői munkafolyamat központibb részévé váljon? E témák megvitatása a dokumentumot egy visszatekintő kritikából egy előre tekintő kutatási programra változtatná, utat mutatva a terület számára.

Végezetül, bár a dolgozat helyesen azonosítja az ipari méretű empirikus tanulmányok hiányát, hasznos lenne egy hipotetikus vagy általánosított esettanulmány beépítése a pontok konkrétabb szemléltetése érdekében. Bár a valós adatok szűkösek lehetnek, egy részletes gondolatkísérlet arról, hogy például egy szeletelő eszköz hogyan bukna el vagy lenne sikeres egy nagyméretű modern alkalmazáson, például egy felhőalapú mikroszolgáltatáson vagy egy összetett mobilalkalmazáson, élénkebbé és érthetőbbé tenné a technikai kritikákat a szélesebb szakmai közönség számára.

Összességében a dolgozat komoly tudományos hozzájárulás, amely újszerű módszertani megközelítést, alapos adatfeldolgozást és nemzetközi kitekintést kínál. Kritikai szempontból elsősorban a fogalmi pontosítás, a hivatkozások aktualizálása, az ábrák és általánosítások gondosabb kidolgozása, a jövőbeli kutatási irányok összefoglalása erősíthetné tovább a munkát.

5. Önálló tudományos teljesítmény értékelése

A teljes életművet tekintve az MTMT több, mint 130 közleményt számlál, kb. 20% folyóirat, 75% konferencia megjelenés, döntő többségük 3 vagy többszerzős. Az MTMT összesítő táblája több, mint 1300 független hivatkozást mutat. Egyenrangú szerzői profilt tekintve és normalizálva a teljesen önálló tudományos hivatkozási pontérték 300 feletti. A google scholar által mutatott 28 h-index és 63 i-10 index felépített életműre utal.

6. Összefoglalás, nyilatkozat

Mindezek alapján kijelentem, hogy a doktori munka tudományos eredményeit elegendőnek tartom az MTA doktori cím megszerzéséhez, és javaslom a nyilvános vita kitűzését.

Budapest, 2025. szeptember 13.

Kovács Attila

Kovács Attila
egyetemi tanár