Ferenc Rudolf

“*Improved Bug Prediction Through Conceptual Metrics and Machine Learning*” című MTA doktori értekezésének bírálata

Témaválasztás

Az értekezés témája a szoftver alkotásokban rejlő hibák előzetes becslése koncepcionális metrikák és gépi tanulás segítségével. A választott téma a mindinkább a társadalmi intelligencia hordozójává váló informatika egyik kulcskérdését, a szoftverminőséget célozza.

Az informatika robbanásszerű elterjedésével ugyanis a szoftverek létrehozása önálló iparággá nőtte ki magát, ugyanakkor minden fontossága ellenére a szoftver minőségbiztosítása ezt a trendet csak erős késéssel képes követni. A szoftverben rejlő hibák hatása ugyanis ugyanakkor mind jelentősebbé válik, hiszem a gazdasági, társadalmi és termelési folyamatok esetében a szoftverben rejlő hibák potenciális kihatása jelentős kockázatokat eredményez. Ennek megfelelően a rejtett hibák nem csak az egyes programok, hanem az azon alapuló szolgáltatások szempontjából is meghatározóak.

Ezzel egyidejűleg a szoftver alkotásoknak mind a mérete, mind pedig a komplexitása folyamatosan nő, sőt a nagy bonyolultságú alkalmazások esetében már nagylétszámú fejlesztő teamek dolgoznak a feladaton, amelyeknek munkája finom részletekig nem hangolható össze.

Az értekezés fő célkitűzése ezen hibák előfordulásának becslése

A szoftver fejlesztési folyamat végeredményeképpen előálló forráskód elemzésével. A jelölt eredményei két fő csoportra oszthatók:

1. A hagyományos, a forráskód strukturális jellemzőin alapuló becslési eljárásokat finomítva - a *koncepcionális kohézió fogalmát* bevezetve- kimutatja, hogy a koncepcionális kohézióból származtatott metrika egy azoktól *független, és a becslés pontosságát javító új metrika* (T'1). Elemzi ennek felhasználását a *hatásanalízis*ben, majd új, a korábbiaknál könnyebben számolható és a strukturális metrikákon alapuló becslést érdemben javító *új koncepcionális metrikákat* dolgoz ki (T2-T3).
2. A második nagyobb csoport a *gépi tanuláson alapuló modellek* kidolgozásához alapvető fontosságú *nyilvános, egységes hiba adatbázist* alakít ki (T4), és ennek felhasználásával *mély tanulási módszerek* segítségével ad módszertant a hibák becslésére (T5).

A célkitűzések ambiciózusak, és jellegüknél fogva *a kísérletes számítástudomány* körébe sorolhatóak.

Ennek megfelelően az elvégzett kísérletek benchmark jellegűek, és az eredményeket a hatékonyságok igazolja, de nincs mód a hagyományos értelemben vett matematikai helyességbizonyításra, csak statisztikai alátámasztásra.

Kiemelendő ugyanakkor, hogy lényeges új eredmény a konkrét metrikák kialakításán túl az azok definícióját és ellenőrzését támogató metodika is..

Az értekezés felépítése

Az értekezés nyolc számozott fejezetből áll, amelyet egy számozott és egy online függelék egészít ki. Az értekezés irodalom hogy jegyzéke rendkívül gazdag, a jelölt saját publikációit is beleértve mindösszesen 248 tételt tartalmaz.

1. fejezet

Ez a bevezető fejezet az értekezés célkitűzését foglalja össze és kiemeli a főbb eredményeket.

2. fejezet

A 2. fejezet rendkívül tömören, mindössze 5 oldal terjedelemben foglalja össze a műszaki és matematikai háttérapparátust. Az összefoglalóban a műszaki háttérismeretek leírása érthető, de lényegesen nagyobb terjedelmet érdemelt volna a statisztikai módszerek és a gépi tanulási technikák ismertetése.

Megjegyzés 1: Az alkalmazott módszerek helyesek, de mindenképpen legalább informális indokolást igényelt volna, hogy *a vizsgált jelenségek milyen tulajdonságai* alapozzák meg azt, hogy a fókusz a lineáris módszereké.

> Például, a korrelációanalízis esetében a Pearson korreláció használata szokásos, de alapvetően a lineáris kapcsolatot tükrözi, ezért számos más asszociáció erősséget vizsgáló metrika is szóba jöhetne. Hasonlóan a főfaktor analízis (PCA) is csak egyike a szóba jövő technikáknak.
>
> A gépi tanulási módszerek esetében néhány kiragadott, de gyakran használt módszerre szorítkozik az ismertetés.

Megjegyzés 2: A kiértékelő módszerek az alapvető statisztikai kiértékelésre szorítkoznak, lényegében a konfúziós mátrix köré csoportosulnak, ugyanakkor a gépi tanulás mögé ma már a a tanulás végeredményét minősítő modellmagyarázatnak és -ellenőrzésnek ennél sokkal bővebb repertoárja jött létre.

Ezen aspektusok tisztázása további jelentős hozzájárulást jelentene az elért, alkalmazott tudományos szempontból jelentős kísérleti eredmények és metodika elmélyítéséhez.

Megjegyzés 3: Miközben elfogadom, hogy a koncepcionális kohézió alkalmazása egy eredeti gondolat, de indokolt lett volna a kisérleteket egybevetni a fejlesztési folyamat és feladat számos paraméterét minőségbecslésre felhasználó klasszikus COQUALMO megközelítéssel (amely egyúttal egy széles faktorlistát is ad a benchmarkok pozicionálásához).

Megjegyzés 4: Hasonlóképpen a viszonylag friss (az értekezés eredményeinél később publikált ISO/IEC 5055:2021 (Information technology — Software measurement — Software quality measurement — Automated source code quality measures) szabvány is potenciálisan releváns.

3. Fejezet

Az értekezés 3. fejezete vezeti be a koncepcionális kohézió fogalmát, majd módszert ad a Latent Semantic Indexing módszer alkalmazásával a szövegkoherencia feltárására és ebből szoftver kohéziós mértékek számítására.

A szorosan kapcsolódó korábbi munkákat a 3.2 alfejezet foglalja össze. Ez az alfejezet túlzottan tömör. A számos hivatkozás jól mutatja ugyanakkor a jelölt ismereteit a területen, de a csoportosítás és az egyes alkalmazott módszerek ismertetése elmaradt. Ahogy a szerző helyesen ismerteti, számos ilyen módszernek létezik közös egyesített keretrendszere, de a szövegből ennek mibenléte sem derül ki. ***Ez a hiányosság meglehetősen nehézzé tette a további részek követését, sőt a bírálat elkészítéséhez meglehetősen sok, külső forrás áttekintését tette szükségessé.***

Jól követhetők viszont a jelölt saját eredményeit bemutató 3.3-3.7 alfejezetek. Az az alapgondolata, hogy a program szövegben elhelyezkedő nem strukturált szövegszerű információk a kód mellett fontos információt hordoznak és alkalmasak a koncepcionálisan jól strukturáltság felmérésére a szokásos jó programozói gyakorlatokat ellenőrző metodika kialakítására. Bár ezekre nincs formális ipari szabvány, de az iparszerű programozásnál ezek alapkövetelmények, és a nagyobb vállalatok a sokszereplős teammunka, az ellenőrizhetőség, nyomonkövethetőség és erkölcsi-műszaki karbantarthatóság érdekében ezeket belső kódolási előírások formájában elő ís szokták írni.

A szerző ezt a gondolatot bontja ki szoftver minőségi metrikává, azáltall, hogy ezeket a szövegszerűen értelmezhető elemeket bevált szöveg kohéziós algoritmus segítségével elemzi, és az egyes osztályokat alapegységnek tekintve azokból szoftver kohéziós metrikákat számít.

Elegánsnak tartom az általános szövegelemzési feladat és a forráskódban rejlő nemstrukturált szöveginformáció összerendelését. A bevezetett definíciók alapvetően tiszták, bár a 3.2 képlet nem tűnik teljesen tökéletesnek de ez nem értelemzavaró.

Miközben az alapgondolatot a jó programozói gyakorlat plauzibilisé teszi, természetszerűleg szükséges annak legalább mintapéldákon keresztüli kipróbálása. Az értekezés két fontos kérdést tesz fel: a javasolt C3 metrika ad-e járulékos információt a strukturális kohéziós metrikához képest, illetve ennek fényében javítja-e az osztályokban rejlő hibák becslését.

Az értekezés három nyitott forráskódú példán vizsgálta meg a módszert gyakorlati alkalmazását. Összességében az összkép kedvező, és mindkét kutatási kérdésre a példák pozitív választ adnak. Az érdekes példamutatóan taglalja azokat a lokális jellemzőket, amelyeknél a várt eredmények eltérnek a globális összképtől.

A 3.6 alfejezet *“Threats to Validity”* foglalja össze azokat a főbb kísérleti hiányosságokat, amelyek az eredmény teljes általánosítását akadályozzák.

Megjegyzés 4: A szerző maga is kiemeli azt, hogy a vizsgálat csak kis mintán történt. Ez azért is sajnálatos, mert a megalapozó publikáció 2008-as.

Megjegyzés 5: Az eredmények érvényességének diszkussziója (itt és a későbbi fejezetekben) műszakilag követhető, de célszerű lett volna egyrészt visszavetíteni a problémára és modelljére, másrészt a torzító hatások kompenzálási lehetőségeit megvizsgálni.

A magam részéről még egy nehézséget látok:

Megjegyzés 6: (Reprezentativitás) A kiértékelés során a minták kiforrott (magas verziószámú) szoftver termékekre vonatkoznak, és az elemzés így jó minőségű kódot vizsgál. Indokolt lett volna kiforratlan, illetve gyenge minőségű kódokra is az elemzés elvégzése ahhoz, hogy a detektálóképesség a másik irányban is látható legyen.

A 3.7 fejezet foglalja össze a kontribúciót és az azokat alátámasztó cikkeket. A szöveg korrektül elhatárolja a rendkívül rangos folyóiratban megjelent fő publikáció társszerzőinek és a jelöltnek a munkáját. Miközben az alapötlet definiálása a társszerzők munkája, a teljes kísérleti validációs metodika a jelölté. Ez utóbbi önálló tudományos eredményként fogható fel, s célszerű ennek általánosított megfogalmazásai is.

4. Fejezet

A fejezet a korábbi szemantikus kohézió fogalmát továbbfejlesztve alkalmazza azokat a rendszerben bekövetkező változások hatásanalízisére. Az alapgondolat az, hogy a rendszer egyes részeinek erkölcsi-műszaki karbantartásának hatása az ahhoz szemantikusan kötődő további komponensekre terjed ki. A hatás terjedelménynek jó becslése alapvető a karbantartási tevékenységben, hiszen ez teszi lehetővé a változtatást követő újraellenőrzés kellően alapos, de egyidejűleg hatékony elvégzését.

A fejezetben elért eredmények algoritmikus alapgondolata az, hasonlóan a tradicionális structurális csatoláson alapuló metrikákéhoz, hogy a szemantikus csatolás meghatározó indikátoraa tovaterjedő hatásoknak.

A 4.2 alfejezet a korábbi kutatások eredményeit foglalja össze. Ezek közül a 4.2.1 szekcióban foglaltak közvetlenül kapcsolódnak az alfejezet mondanivalójához, a 4.2.2 viszont olyan metodikai háttérmódszereket ismertet, amelyek akár egy globális korábbi fejezetbe is beilleszthetőek lennének a 3.3 alfejezetben foglaltakkal egységesítve.

A 4.3 fejezet a koncepcionális csatolási metrikákat finomítja, majd a 4.4-4.6 alfejezetek egy példán keresztül bemutatják a finomított definíciók alkalmazását hatásanalízisre.

Megjegyzés 7: A korábbi alfejezethez fűzött megjegyzések itt különösen érvényesek, van-e további alátámasztó kisérleti eredmény?

5.fejezet

Ez a fejezet lényegesen elmélyít a korábban bevezetett metrikákat és alkalmazástechnikájukat. Egyfelől egy küszöbmetrikát bevezetve kiszőri a gyenge koncepcionális csatolásokat, megkönnyítve és egyes strukturálisékhoz közelítve a koncepcionális metrikák származtatását, valamint javítja a metrikák származtatását a preprocesszálás finomításával (stemming).

A másik ágon 61 metrika elemzésével gépi tanulási módszerek és logisztikai regresszió segítségével vizsgálja azok szoftver hiba előrejelző képességét.

Az elsö csoportba tartozó eredmények Alapvetően a szövegfeldolgozás témaköréhez kapcsolódnak ismert módszerek alkalmazását jelentik a minőségellenőrzés célterületén. A kísérletek tanulsága szerint hatékonyak.

Megjegyzés 8: A dísszertáció abból -a természetesen számos esetben realisztikus-, feltételezésből indul ki, hogy a vizsgálandó szoftver alkalmazásból csak a kód áll rendelkezésre. Felvetődik azonban az a kérdés, hogy, ha rendelkezésre áll egyéb projektdokumentáció is, ezen járulékos információforrásoknak mi a hatása a metrikákra, és azok alkalmazására. Például modellvezérelt tervezés előkészítésére tradícionálisan szokásos egy célterületi szótár majd abból származtatott hierarchikus taxonómia kialakítása. Ilyen esetben (feltételezve a fejlesztési folyamat kellő fegyelmezettségét) nem a koncepcionális fogalmak visszaállítása a feladat, hanem a meglevő taxonómia integrálása a minőségmérési folyamatba.

Megjegyzés 9: A modelszármaztatási feladat esetében alapvető nehézséget jelentett a megértésben az egyes módszerek definíciójának hiánya, amit részlegesen pótolt a hivatkozott online függelék illetve a későbbi 6.4 alfejezet). Mindenképpen indokolt lett volna a megvizsgált modellek kiválasztásának rövid indoklása illetve az eredmények legalább kvalitatív érvelés szintű interpretációja. Ez különösen igaz azokra az esetekre, amikor a modellek jól magyarázhatóak (például döntési fák). Egy ilyen elemzés lényegesen mélyebb megértéshez vezethet.

Megjegyzés 10: A kidolgozott modellek és az ahhoz azokhoz kapcsolódó metrikák hatékonyságának összehasonlítása dominánsan nagy globális adatkészleteken alapul. Az ilyen nagyméretű és potenciálisan inhomogén adatkészleteknél "kiátlagolódnak" a különféle minőségű tartományokra vonatkozó metrikák (például csoportmunka esetén, ha nincsenek feszes előírások, a kommentezés az egyes egyedi fejlesztőktől függ). Ugyanakkor az intuíció azt mondja, hogy a "legjobb" metrika kiválasztása erősen függhet a vizsgált szoftverobjektum tulajdonságaitól. Például egy gazdagon és értelmesen kommentezett kód a működés alapján jobban támogathatja a koncepcionális metrikákat. Néhány egyszerű jellemző előzetes megállapítása útmutatást nyújthat a legjobb prediktor megtalálásához.

6. fejezet

Ez a fejezet az empirikus kiértékelési metodikák egyik alapproblémájára ad elegáns és korszerű megoldást adni a szoftverminőség szűkebb területén, mégpedig a különféle módszerek hatékonyságának kiértékelésére és összehasonlítására szolgáló benchmark adatkészletekét.

A probléma fontosságát jól jellemzi, hogy például a mesterséges intelligencia fő sodorvonalában, a képfelismerésben használt alapvető mintakészletről is kiderült, hogy a képek jelentős része félrecímkézett. Érdemes ugyanakkor megjegyezni azt is, hogy ez a vizsgálat és az értekezés szerzőjének tevékenysége időben párhuzamosan zajlott, jelezve a díszszertációban közölt eredmények fontosságát és újszerűségét.

Maga a fejezet számos külső hivatkozást tartalmaz, tekintettel arra ugye a forrásként felhasznált adatbázisok még terminológiailag sem egységesek.

A fejezet a problémát magas színvonalon oldja meg és több aspektusból is elemzi a benchmarkok illetve a metrikák alkalmazásának hatását. Ezzel a felhelyezettel kapcsolatos kritikai megjegyzésem alapvetően a szerkesztést érinti, számos értekezés fűszövegében nem való színűleg technikai részlet helyett lényegesen jobban ki kellett volna emelni az általános munkafolyamatot, ideértve a terminológiai és definíciós egység megteremtését, indokolni a vizsgálati módszertant és a kísérleteket tömören ismertetni.

Kiemelendő, hogy az egységes hiba adatbázis a nemzetközi kutatások támogatásához, ideértve a különféle új eredmények kiértékelését is, alapvető fontosságú, ezért is szerencsés nyitottá tétele.

7.fejezet

A fejezet a hibákat előrejelző mély neurális háló kidolgozását célozza. A közvetlen célkitűzésen túl expliciten ambícionálja egy teljes munkafolyamat megalapozását is. A rövid bevezetés után áttekinti a vonatkozó irodalmat (az itt ismertetett megközelítések egy részét korábban is el lehetett volna mondani). Maga a tanítási folyamat korrekt, és a jó adatmérnöki gyakorlatnak felel meg. Alapos az előfeldolgozási fázis és az egyes lépések kiértékelése is.

A globális metrikák szempontjából a fejezet zárt.

Megjegyzés 11: Hasonlóan a korábbi fejezethez fűzött megjegyzésekhez, itt is indokolt lett volna a kiadódó eredmények interpretálása. Specifikusan kínálja magát a modell mélyebb megértéséhez a magyarázó modellanalízis (explanatory modell analysis –pl https://ema.drwhy.ai/), amelyből értékes hatás analízis nyerhető, konstruktív szoftverfejlesztési metodikai következtetésekkel..

Összefoglalás

Éppen az értekezés újszerűsége miatt számos koncepcionális kérdés is felvetődik:

1. **Mennyiben alkalmasak az értekezésben feltüntetett módszerek tetszőleges esetben a szoftverben rejlő hibák számának becslésére. A vizsgált benchmarkok (pl. Mozilla) elég nagyok ahhoz, hogy az adott minta alapján statisztikailag releváns következtetéseket lehessen levonni. Azonban az ismertetett benchmarkok meglehetősen kiforrottak, jelentős és fegyelmezett fejlesztői bázissal.**
    a. **Hogyan viselkedik ez a metrika a szoftvert gyenge fejlesztői bázison és például a szokásos kommentezésre és névadásra vonatkozó előírásokat nem teljesítő esetben?**
    b. **Mik a módszer alkalmazhatóságának befolyásoló faktorai?**
    c. **Van-e alkalmazási küszöbszint?**
2. **Kritikus szoftverek esetén szokásos követelmény a nyomonkövethetőség, azaz a kódnak dokumentáltan kell tartalmaznia az eredeti követelményrendszerrel való összerendelést. Felhasználható-e ez az információ a koncepcionális csatolás fogalmának magasabb szintű aspektusokra való kiterjesztésére?**
3. **A modellalapú szoftvertervezés jó gyakorlatának alkalmazása esetén a koncepcionális és strukturális tagolás erősen közeledhet. Ilyen esetben mi a viszonya az értekezésben vizsgált koncepcionális és strukturális metrikáknak, fennmarad e a koncepcionális metrikák erős kiegészítő ("ortogonális") jellege?**
4. **A kidolgozott koncepcionális kohézió alapú módszerek mennyiben alkalmasak tradicionális módon kidolgozott szoftverek mikroszolgáltatás alapúvá átdolgozásának támogatására?**
5. **Megfigyelhető-e, hogy egy-egy szoftver megoldás tisztázása például refaktorálása során a koncepcionális csatolási metrikák javulnak, Illetve mennyiben alkalmasak a metrikák ilyen folyamatok heurisztikus vezérlésére? Ilyen esetben a szoftvermérnöki intuíció szerint jó minőségű tervekben a koncepcionális és strukturális felépítés közel esik. Mennyiben maradnak ilyenkor a koncepcionális és strukturális metrikák viszonylag korrelálatlanok?**
6. **A gyakorlati alkalmazások számára izgalmas az eredményeknek a fejlesztési modellbe integrálása. Milyen ajánlásokat javasol?**
7. **Az értekezésben bemutatott eredmények alapjainak kidolgozása óta a mesterséges intelligencia eszköztára drasztikus fejlődésen ment át. Az eredmények továbbfejlesztése szempontjából ez milyen új algoritmikus és technológiai lehetőségeket jelent?**

Az értekezést alapos irodalomkutatás alapozza meg. Az irodalomjegyzék 248 tételt tartalmaz, amelyből 27 a jelölt saját publikációja. A technikai jellgű hivatkozások zöme napra kész. A külső tudományos hivatkozások igen alaposak, bár az utolsó néhány év feldolgozása esetleges.

Az értekezés fő érdeme egy problémakör teljes mélységű végig elemzése. A disszertációt megalapozó kutatómunka színvonalas minősége mellett mennyiségében is számottevő, Ennek megfelelően az értekezés maga túlzsúfolt.

A célterület számára fontos, és gyakorlati szempontból kiemelten jelentős megoldások halmazát kínálja, a kutatás- fejlesztés implementációig és kísérleti validálásig terjedő teljes spektrumát vé-

gigvive tudományos megalapozottsággal. Az értekezést megalapozó kutatómunka nemzetközileg is jelentős, több nemzetközi kooperatív projekt eredménye. Egyes metodikai elemeinek és eredménytermékeinek jelentős a nemzetközi kihatása, és nemzetközileg is úttörőnek tekinthetők.

Kiemelendően hasznos a nemzetközi kutatói közösség számára a hiba adatbázis és a metrikák egységes szemléletű rendszere.

Az értekezésben a prezentáció súlypontozása nem kellően arányos. Túl sok hangsúlyt kap a validációs kisérletek technikai részletezése (részben olyan információtartalommal, ami csak részlegesen releváns), Az adott terjedelmi korlátok mellett így túl kevés jut az általánosabb szintnek.

Ebből a szempontból példamutató a tézisfüzet.

Az értekezés értelmezését a nagy anyag mellett helyenként nem önhordó volta meglehetősen megnehezíti. Az értekezésnek nemcsak az alapját képezik rangos cikkek, de a prezentáció is közel van a szűk értelemben szakterületen szokásoshoz, de nem ad a “kívülről érkező” szakembereknek megfelelő belépési küszöböt.

Az értekezésben szerencsés lett volna az irodalmi hivatkozások részletesebb kifejtése, akár az egyes benchmark kisérletek technikai jellegű részleteinek függelékbe helyezésével.

Az értekezés angol nyelven íródott, nyelvi minősége kiváló. A technikai jellegű ábrák gondos kivitelűek. Szerencsés lett volna ugyanakkor, ha az értekezés többi szövegközi magyarázó ábrát illetve a kidolgozott algoritmusokat bemutató pszeudokódot is tartalmazott volna, hiszen a kidolgozott metodikák jelentős szellemi értéket képviselnek, így akár önállóbb formában való megjelenítésük is indokolt lett volna.

Az értekezés szövegének jobb tördelése és konzekvens tipográfiai kiemelések ugyancsak javítottak volna a követhetőségen.

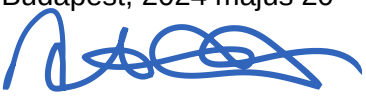
Összefoglalásként elmondható, hogy Ferenc Rudolf értekezésében egy, a gyakorlat számára kiemelten fontos problémakörre, a szoftverminőség becslésére adott újszerű megoldást, amely a kidolgozás fázisában kifejezetten előremutató matematikai modellezést, szélesspektrumú algoritmikát és eszköztárat alkalmazott. Módszerei alapvetően helyesek és számos ponton továbbfejlesztést inspirálnak.

Az eredményeket jelentős gyakorlati szoftvermérnöki alkotó munka által megalapozott kiterjedt kísérleti vizsgálatok igazolják, bár sok értékes metodikai eredmény csak informálisan jelenik meg a szövegben. A munka jellegénél fogva a tudományos eredmények alátámasztása csak empirikus lehet, ehhez értékes és időtálló megközelítést mutatott be és eszközöket dolgozott ki.

A publikációk rangosak és a kontribúciókat alátámasztják. A cikkek zöme társszerzős, de ezt a megoldandó kutatási kérdések komplexitása és a kísérleti jelleg teljességgel indokolja.

A fentiek alapján az értekezés nyilvános vitára bocsátását és sikeres védés esetén pedig az MTA doktora cím odaítélését javaslom.

Budapest, 2024 május 20



Patricza András

MTA doktora