ORODJE PROMIS IN NJEGOVA UPORABA PRI ANALIZI PROGRAMOV

Janez Brest, Peter Kokol, Marjan Mernik, Viljem Žumer
Fakulteta za elektrotehniko, računalništvo in informatiko
Univerza v Mariboru, Smetanova 17, 62000 Maribor, Slovenija
janez.brest@uni-mb.si

POVZETEK

Osnovno načelo programskega inženirstva je razvoj kakovostne programske opreme. Najobičajnejša pot za dosego kakovostnih programskih produktov, je uporaba metrik in standardov. Uporaba programskih metrik temelji na analizi programov, za kar potrebujemo učinkovite analizatorje. Zaradi slabosti običajnih analizatorjev, smo v našem modelu razvili in uporabili orodje PROMIS, ki temelji na generatorjih prevajalnikov. Orodje PROMIS omogoča analizo programov na osnovi standardnih metrik in novih fraktalnih metrik.

ABSTRACT

A fundamental goal of software engineering is the development of high quality software at low cost. There are various definitions of software quality and various ways how to achieve it. The most usual approach is implementation of standards and metrics, which emphasizes the importance of program analysis tools. Conventional analysers have several limitations and are not easy to design, implement and maintain. To overcome these weaknesses we have introduced a new approach to program analysis, namely using compiler compilers as a generator of program analysis tools. We have successfully used PROMIS in the program analysis using conventional software metrics and new fractal metrics.

■■■

Uvod

Analiza programov je pomembna za razumevanje obnašanja programov in značilnosti njihovega razvoja [6]. **Osnovno načelo programskega inženirstva je razvoj visoko kakovostne programske opreme. Orodja, ki so namenjena analizi programov, potrebujejo pisci programov za določevanje kritičnih delov programa, načrtovalci arhitektur za ugotavljanje lastnosti novih arhitektur, pisci prevajalnikov za ugotavljanje njihove učinkovitosti, vodje projektov za določanje značilnosti procesa razvoja, zanesljivosti programov, učinkovitosti programerjev, izračun programskih metrik, itd. V tem članku bomo opisali orodje PROMIS ("Program Metrics Support"), ki smo ga realizirali na sistemu UNIX** [5]. **Orodje PROMIS omogoča analizo programov na osnovi standardnih metrik in novih fraktalnih metrik** [1,2].

Z uporabo modernih orodij zelo poenostavimo oblikovanje analizatorjev in njihovo vzdrževanje, skrajšamo čas razvoja in s pomikom na višji abstraktni nivo (regularni izrazi in kontekstno proste gramatike) naredimo analizatorje bolj pregledne in razumljive tudi neprogramerjem.

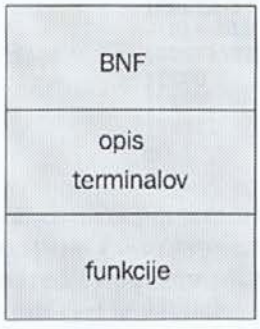
Z orodjem PROMIS smo zgradili analizator in ga uporabili pri analizi programov s pomočjo fraktalnih metrik.

Orodje PROMIS

Orodje PROMIS avtomatsko tvori analizator programov. Orodju na vhodu podamo datoteko, ki vsebuje Backus-Naurovo formo BNF, izhod pa je analizator programov (slika 1).

Slika 1: Delovanje orodja PROMIS

Orodje PROMIS na vhodu zahteva datoteko z naslednjo obliko:



Slika 2: Oblika vhodne datoteke

Vhodna datoteka (slika 2) je sestavljena iz več delov. Najpomembnejši je vsekakor del, ki vsebuje zapis slovnice s pomočjo BNF.

Orodje PROMIS omogoča, da terminalne simbole opišemo z regularnimi izrazi. Če bi na primer želeli terminalni simbol `while` napisati kot `WHILE, While, WhILe` itd., ga lahko definiramo takole:

```
while    [Ww][Hh][Ii][Ll][Ee]
```

V prvem delu vhodne datoteke zapišemo BNF v obliki, kot jo na vhodu pričakuje orodje PROMIS. V drugem delu definiramo terminale, kakor smo prikazali na primeru terminalnega simbola `while`. V zadnjem delu so funkcije, ki so napisane v programskem jeziku C. Uporabljamo jih takrat, kadar želimo, da analizator, ki ga tvori orodje PROMIS, opravlja dodatne naloge.

Analizator, ki ga tvori orodje PROMIS, deluje kot razpoznavalnik. Nastane kot produkt orodja *YACC* (Yet Another Compiler Compiler) [3,6] in orodja *lex* [3,4,6]. Pregledovalnik vhodni tekst razbije v atome, ki so vhod razpoznavalniku. Razpoznavalnik preverja sintaktično pravilnost vhodnega teksta. Ko razpozna atom, izvede ustrezno akcijo.

Delovanje orodja PROMIS

Oglejmo si, kako deluje orodje PROMIS. Orodje avtomatsko tvori analizator na naslednji način:

1.) tvori datoteko, ki vsebuje podatke o analizi,
2.) transformira BNF v datoteko, ki služi kot vhod orodju *YACC*,
3.) na podoben način tvori tudi datoteko, ki je vhod orodju *lex*,
4.) na koncu izvede še prevajanje in povezovanje.

Poleg BNF, definicij terminalov in funkcij moramo orodju povedati, kaj naj analizira. Zanima nas npr. število vseh besed v tekstu, kolikokrat se ponovi določena beseda, koliko parametrov imajo funkcije, itd. Orodju v posebni datoteki naštejemo vse, kar nas pri tej analizi zanima.

Oglejmo si to na primeru. Pri večini programskih jezikov z oklepaji (okrogla oklepaja "(" in ")") spreminjamo prioriteto računanja pri aritmetičnih izrazih. Zanimajo nas vse takšne situacije in zato želimo prešteti vse oklepaje, ki nastopajo v aritmetičnih izrazih. Ne moremo preprosto prešteti vseh oklepajev, saj oklepaji nastopajo še drugod. Če vzamemo na primer programski jezik C, oklepaje uporabljamo tudi pri funkcijah (argumenti funkcij), v logičnih izrazih, pri spreminjanju tipov in morda še kje. Rešitev problema ni preprosta, vendar lahko z orodjem PROMIS učinkovito rešujemo tudi takšne probleme, saj smo ga razvili prav z namenom, da z njim tvorimo analizatorje, ki jih ni možno preprosto napisati v enem od programskih jezikov.

Izhod, ki ga dobimo, je izvršljiva koda - analizator.

Analiza programov s pomočjo fraktalnih metrik

Teorija kaosa in fraktali [10, 13] so v zadnjih letih postali ena izmed najzanimivejših raziskovalnih tem. Z njihovo pomočjo lahko razložimo delovanje mnogih naravnih in umetnih sistemov z mnogih različnih področjih. Osnovna značilnost fraktalov je samopodobnost, kar pomeni, da so invariantni na spremembe merila, razne transformacije ipd. Najenostavnejša definicija fraktalov je, da so sestavljeni iz delov na nek način podobnih celoti in njihov najpomembnejši atribut je njihova dimenzija (fraktalna dimenzija - D).

Glede na splošno uporabnost fraktalov v praktično vseh znanstvenih področjih smo jih poskušali uporabiti tudi pri analizi programov. Razvili smo naslednje fraktalne metrike:

- Zipfovo,
- imena spremenljivk,
- daljnosežne korelacije.

V članku prikazujemo primer, in sicer metriko imena identifikatorjev [2,4].

Imena spremenljivk

Imena spremenljivk v računalniških programih lahko v splošnem opišemo z naslednjim regularnim izrazom:

Črka, Številka, Podčrtaj (Črka, Številka, Podčrtaj)*

iz katerega sledi, da je ime spremenljivke invariantno na:

1. *vrtenje* - črke v imenu spremenljivke lahko zavrtimo in rezultat vrtenja je spet ime spremenljivke;
2. *deljenje* - če ime spremeljivke razdelimo na enega ali več delov, je tudi vsak tako dobljen niz ime spremenjivke;
3. *združevanje* - če združimo eno ali več imen spremenljivk, je dobljen niz spet ime spremenljivke.

Zaradi invariance na zgornje transformacije, lahko spremenljivko obravnavamo kot fraktal. Nize dobljene iz zgornjega regularnega izraza lahko tako modeliramo kot Cantorjevo množico [2], katere fraktalno dimenzijo lahko izračunamo z enačbo:

$$D = \frac{1}{1 - \frac{\log(1 - p_0)}{\log N}} \qquad (1)$$

kjer je N število različnih črk uporabljenih v besedah in p_0 verjetnost uporabe omejevalnih simbolov.

Fraktalne metrike in kakovost programske opreme

Relacijo med fraktalnimi metrikami in kakovostjo programske opreme smo analizirali na mnogih primerih. Naključno smo izbrali 50 študentov in jim dali nalogo, ki so jo rešili v programskem jeziku *fortran*. Analizirali smo fraktalno metriko D in znane konvencionalne metrike (indirektne meritve) ter jih primerjali z nekaterimi atributi, ki na nek način označujejo kvaliteto programov, kot so čas načrtovanja in pisanja programov, število verzij, ocena, ki jo je študent prejel za opravljeno delo, uspeh študenta v srednji šoli in njegova ocena pri računalniških predmetih v srednji šoli (direktne meritve). Dobljene rezultate 33-ih študentov, ki so uspešno rešili nalogo, prikazuje tabela 1. Slika 3 prikazuje Pearsonove korelacijske koeficiente med fraktalno metriko D in indirektnimi meritvami. Na sliki 4 pa so prikazani Pearsonovi korelacijski koeficienti med indirektnimi meritvami (metrikami). Izračuni so bili narejeni s pomočjo programskih paketov za delo s statistiko.

Če natančno pogledamo tabelo 1 in sliki 3 in 4, opazimo, da je večina korelacijskih faktorjev relativno majhnih, vendar so nekateri korelacijski faktorji med direktnimi meritvami in fraktalnimi metrikami v signifikantnem razredu 10%. Vidimo, da fraktalne metrike mnogo bolje ocenjujejo verzijo in oceno programa kot konvencionalne. Prav tako vidimo, da so korelacijski faktorji med konvencionalnimi in fraktalno metriko majhni, kar pomeni, da fraktalne metrike merijo druge lastnosti programov kot konvencionalne.

Uporaba orodja PROMIS

Področja uporabe so naslednja:

1. analiza programov
   - konvencionalne in fraktalne metrike,
   - ugotavljanje kakovosti in kvalitete programov,
   - ocenjevanje programov (npr. študentski programi),
   - statična analiza (dekompozicija programov v programske grafe - izvajanje na večprocesorskem računalniku),
2. jezikoslovje,
3. itd.

Prednosti PROMIS-a so:

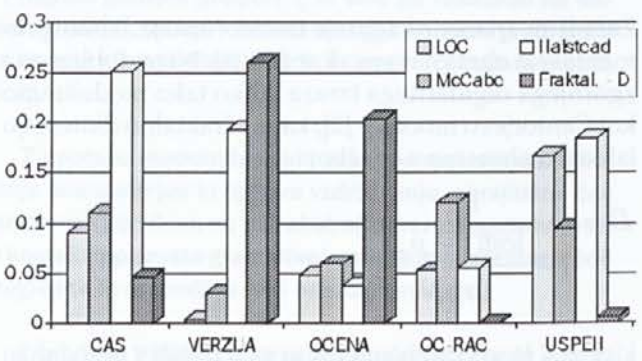
- enostavna uporaba orodja,
- neodvisnost od programskega jezika,
- uporaba standardnih zapisov BNF oz. EBNF,
- interna kontrola vhodnih vzorcev (razpoznavanje),
- učinkovitost; rešljivi so vsi problemi, kjer lahko vzorce za analizo opišemo z (E)BNF,
- hitrost je več kot zadovoljiva,
- možnost razširitve orodja - dodajanje novih funkcij (semantične akcije).

Zaključek

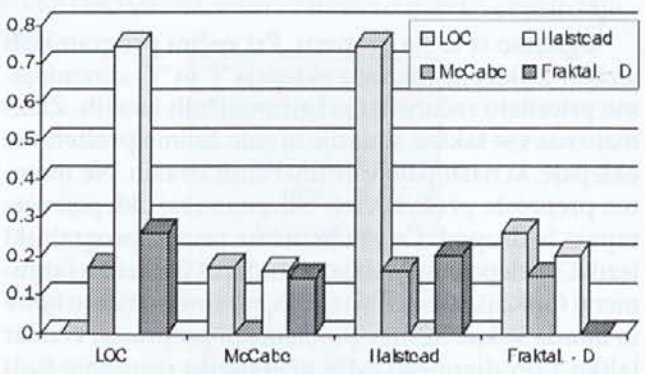
Namen tega članka je bil prikaz uporabe orodja PROMIS, ki temelji na orodjih *YACC* in *lex*. Orodje PROMIS, ki avtomatsko tvori analizator programov, je v prototipni fazi in bo potrebo vložiti še nekaj truda, da bo postalo komercialno orodje.

Orodje PROMIS smo do sedaj uporabljali tako pri analizi študentskih, kot profesionalnih programov. Primerjali smo konvencionalne in fraktalne metrike, ki jih raziskujemo [1,8]. Programi so bili napisani v programskem jeziku fortran, pascal in C. Nameravamo pa izvesti tudi analizo programov napisanih v jeziku visual basic, C++ in Clipper.

Statistična primerjava fraktalnih metrik s konvencionalnimi metrikami kaže, da so fraktalne metrike komplementarne s konvencionalnimi metrikami in zato merijo drugačne programske atribute. Statistični testi nad dobljenimi rezultati in primerjave empiričnih meritev, ki so bile narejene med eksperimentom, kažejo, da so ti atributi v tesni povezavi s kakovostjo programov. Zatorej, fraktalne metrike predstavljajo osnovo za pomembne programske značilnosti, ki smo jih odkrili, vendar njihovega natančnega pomena trenutno še ne poznamo.



Slika 3. Indirektne meritve - direktne meritve



Slika 4. Fraktalna metrika D - konvencionalne metrike

Tabela 1. Direktne in indirektne metrike (Čas - čas razvoja programov, Ver - število verzij, O - ocena programa, U - uspeh v srednji šoli, OR - povprečna ocena računalniških predmetov v srednji šoli, LOC - število vrstic programa)

DIREKTNE METRIKE					INDIREKTNE MERITVE			
Čas	ver	O	U	OR	LOC	McCabe	Halstead	Imena
120	39	9	4	5	94	5	143306	0.865231
135	33	6	4	4	40	2	40608	0.793324
105	1	6	5	5	57	6	114165	0.873196
105	14	6	3	3	55	3	36843	0.882412
60	9	6	2	3	50	2	33975	0.829128
90	42	6	3	2	39	5	12152	0.853917
90	47	8	3	3	51	2	28477	0.803905
105	40	7	5	5	36	4	25363	0.866015
45	15	10	5	4	32	2	27865	0.827969
40	23	10	5	4	88	4	182010	0.898221
135	1	7	3	3	51	3	168072	0.89286
135	21	10	3	3	43	4	121480	0.864104
150	11	6	2	3	51	4	176402	0.858746
120	28	8	5	5	62	4	146348	0.876053
120	10	6	2	3	40	3	14282	0.877913
120	17	6	4	4	32	3	10646	0.882387
30	17	10	5	4	25	4	11272	0.874753
90	9	9	4	4	39	4	57544	0.73735
45	7	10	5	4	40	5	37986	0.73735
135	29	7	3	3	37	5	17124	0.849642
135	45	6	3	3	41	4	33224	0.84132
60	14	10	4	4	36	3	11005	0.876917
120	4	6	3	4	43	6	26799	0.887532
30	9	10	5	5	21	4	13117	0.907395
105	20	9	4	4	29	3	18238	0.900903
135	63	9	3	3	23	4	9731	0.770969
120	19	6	3	3	29	4	18345	0.875095
90	33	8	3	3	27	5	20689	0.766121
90	33	9	4	4	30	3	26962	0.783282
120	21	8	4	5	39	3	81962	0.721057
120	6	7	3	4	33	3	66290	0.820767
90	14	8	4	4	31	3	12387	0.774292
60	41	10	4	4	38	3	41514	0.776728

Literatura

[1] P. Kokol, J. Brest, M. Mernik, V. Žumer, Fractal program metrics: a new way to measure the characteristics of computer programs, Computational methods and experimental measurement VII, *Proceedings of the Seventh International conference on computational methods and experimental measurements (CMEM '95)*, Capri, Italy, May 1995, Str. 41-48.

[2] P. Kokol, V. Žumer, J. Brest, M. Mernik, PROMIS: Software Metrics Tool Generator, *ACM SIGPLAN Notices*, Volume 30, No. 5 May 1995, Str. 37-42.

[3] J. R. Levine, T. Mason, D. Brown, *Lex & Yacc*, O'Reilly & Associates, Inc. Second Edition, USA, Oct. 1992.

[4] M. E. Lesk and E. Schmidt, Lex - A Lexical Analyzer Generator, Technical report, Bell Laboratories, Murray Hill, New Jersey, July 1975.

[5] J. Peek, T. O'Reilly, and M. Loukides, *UNIX Power Tools*, O'Reilly & Associates, Inc. Random Haus, Inc, USA, 1993.

[6] J. Brest, P. Kokol, M. Mernik, V. Žumer, Generatorji prevajalnikov in njihova uporaba pri analizi programov, *Zbornik tretje Elektrotehniške in računalniške konference ERK'94*, Zvezek B, str. 35-38 - Portorož, Slovenija, 26.-28. september 1994.

[7] B. Stiglic, M. Heričko, I. Rozman: How to Evaluate Object-Oriented Software Development? *ACM Sigplan Notices*. Vol. 30, No. 5, pp. 3-10, 1995.

[8] P. Kokol, J. Brest, M. Mernik, V. Žumer, Automatic Generation of Software Quality Analysis Tools - The Case of Fractal Metrics, *Proceedings of the 4th Software Quality Conference*, Dundee, Scotland, UK, July 1995, Str. 423-432, ISBN 1-899796-00-2.

[9] Scroeder M., Fractals, Chaos, Power Laws - Minutes from an infinite paradigm, Freeman, 1991.

[10] Peitgen R. et al, Chaos and Fractals - New frontiers of Science, Springer Verlag, 1993.

[11] Kokol P., Searching For Fractal Structure in Computer Programs, SIGPLAN 29-3, mar. 1994.

[12] Srivastava A., Eustace A., ATOM: A system for building customized program analysis tools, SIGPLAN 29-6, 196-205, jun. 1994.

[13] Bunde A., Havlin S., 1994, Fractals in science, Springer Verlag.

[14] Conte S. D., Dunsmore H. F., Shen V. Y. 1986, *Software engineering metrics and models*, Benjamin/Cummings.

[15] Vidgen R. T., A. T. Wood Harper 1993, Establishing and managing a Relevant Notion of Quality. *Proceedings of SQM 93*, Elsevier 1993, 117-132.

◆

Janez Brest, dipl. ing. rač., je zaposlen na univerzi v Mariboru, oddelek FERI, kot stažist-asistent. Diplomiral je leta 1995 in je trenutno vpisan na podiplomski študij. Raziskovalno se ukvarja s programskimi jeziki, analizo programov in generatorji prevajalnikov. Je avtor večih člankov objavljenih v zbornikih mednarodnih konferenc in mednarodnjih revijah ter je član IEEE.

◆

Dr. Peter Kokol je zaposlen na univerzi v Mariboru, oddelek FERI, kot docent s področja računalništva. Doktoriral je leta 1992. Raziskovalno se ukvarja s programskimi jeziki, analizo programov, razvojem informacijskih sistemov in teorijo sistemov. Je avtor mnogih člankov objavljenih v zbornikih mednarodnih konferenc in mednarodnjih revijah ter je član IEEE, ACM, ISCA in Slovenskega društva Informatika.

◆

Mag. Marjan Mernik je zaposlen na univerzi v Mariboru, oddelek FERI, kot asistent. Magistriral je leta 1994 in pripravlja svojo doktorsko disertacijo. Raziskovalno se ukvarja s programskimi jeziki, posebno z njihovo semantiko, analizo programov in generatorji prevajalnikov. Je avtor večih člankov objavljenih v zbornikih mednarodnih konferenc in mednarodnjih revijah ter je član IEEE, ACM in Slovenskega društva Informatika.

◆

Prof. Viljem Žumer je zaposlen na univerzi v Mariboru, oddelek FERI, kot redni profesor in je vodja Inštituta za računalništvo. Doktoriral je leta 1983. Raziskovalno se ukvarja s programskimi jeziki in računalniškimi arhitekturami. Je avtor mnogih člankov objavljenih v zbornikih mednarodnih konferenc in mednarodnjih revijah ter je član IEEE in Slovenskega društva Informatika.