Métricas de Software para Predecir el Rendimiento de los Algoritmos de Estimación de Distribuciones en la Generación de Casos de Prueba - Descripción de la Base de Datos

R. Sagarna y J. A. Lozano

I. Introducción

El presente documento sirve de complemento al documento referido en el título [4]. En este trabajo, se plantea un problema de clasificación supervisada para predecir el rendimiento de un generador de casos de prueba basado en EDAs [3], para el cubrimiento de ramas. En la base de datos utilizada, los casos corresponden a funciones C [2], y el conjunto de variables está formado por varias métricas de software y el nivel de cubrimiento obtenido por el generador basado en EDAs.

II. Descripción de la base de datos

A continuación se listan los nombres de las funciones tomadas como instancias en la base de datos. Las funciones eliminadas en el paso de preprocesamiento (Sección VI.A en [4]) se marcan en negrita.

Las funciones son las siguientes: ***adi***, *avevar*, *badluk*, ***bcucof***, *bcuint*, *bessi*, *bessi0*, *bessi1*, *bessj*, *bessj0*, *bessj1*, *bessk*, *bessk0*, *bessk1*, *bessy*, *bessy0*, *bessy1*, *betacf*, *betai*, *bnldev*, *caldat*, *cel*, *chder*, *chebev*, *chebpc*, *chint*, *chsone*, *chstwo*, *cntab1*, *cntab2*, *correl*, *cosft*, *crank*, *ddpoly*, ***des***, *eigsrt*, *el2*, *elmhes*, *erf*, *erfc*, *erfcc*, *eulsum*, *evlmem*, *factln*, *factrl*, *fit*, *fixrts*, *fleg*, *flmoon*, *four1*, *fpoly*, *ftest*, *gamdev*, *gammln*, *gammp*, *gammq*, *gasdev*, *gauleg*, ***gaussj***, *gcf*, *gser*, ***hqr***, ***hunt***, *indexx*, *irbit2*, ***jacobi***, *julday*, *kendl1*, *kendl2*, *kstwo*, *laguer*, *locate*, *ludcmp*, *mdian1*, *mdian2*, *memcof*, *moment*, *mprove*, *pcshft*, *pearsn*, *piksr2*, *piksrt*, *plgndr*, *poidev*, *polcoe*, *polcof*, *poldiv*, *polin2*, *polint*, *probks*, *qcksrt*, *qroot*, *ran0*, ***ran1***, *ran2*, *ran3*, ***ran4***, *rank*, *ratint*, *realft*, *rofunc*, *shell*, *simp1*, *simp2*, *simp3*, *sinft*, *sncndn*, ***solvde***, *sort*, *sort2*, *sort3*, ***sparse***, *spear*, *splie2*, *splin2*, *svbksb*, ***svdcmp***, *toeplz*, *tptest*, *tqli*, ***tred2***, *tridag*, *twofft*, *vander* y *zroots*.

A continuación se describen las métricas de complejidad del software usadas como atributos en la base de datos. Se han agrupado de acuerdo al paquete software de métricas del cual se obtuvieron sus valores. Las métricas eliminadas en el paso de preprocesamiento (Sección VI.A en [4]) se marcan en negrita.

Intelligent Systems Group, Universidad del País Vasco, 20018 San Sebastián, España. E-mail: ccbsaalr@si.ehu.es , ja.lozano@ehu.es .

Software: UnderstandC++

Métricas:

Cyclomatic complejidad ciclomática
CyclomaticModified complejidad ciclomática modificada
CyclomaticStrict complejidad ciclomática estricta
CyclomaticEssential complejidad ciclomática esencial
Lines número de líneas en la función
LineBlank número de líneas en blanco en la función
LineCode número de líneas de código fuente en la función
LineComment número de líneas con un comentario en la función
FileLines número de líneas en el fichero conteniendo la función
FileLineBlank número de líneas en blanco en el fichero conteniendo la función
FileLineCode número de líneas de código fuente en el fichero conteniendo la función
FileLineCodeDecl número de líneas con código declarativo en el fichero conteniendo la función
FileLineCodeExe número de líneas con código ejecutable en el fichero conteniendo la función
FileLineComment número de líneas con un comentario en el fichero conteniendo la función
LineInactive número de líneas inactivas para el preprocesador
LinePrep número de líneas de preprocesamiento
CountStmt número de sentencias declarativas más ejecutables en el código fuente
CountStmtDecl número de sentencias declarativas en el código fuente
CountStmtExe número de sentencias ejecutables en el código fuente
CommentToCode cociente del número de líneas con un comentario y el número de líneas de código fuente

Las métricas relacionadas con la Complejidad Ciclomática (CC) se basan en la definición de McCabe [1]. La CC de un grafo de flujo de control G, con m arcos y n vértices, es $m-n+2$; se refiere al número de caminos independientes que atraviesan G. En la CC modificada se cuenta uno para los l arcos de un vértice representando una decisión con varias salidas (semigrado positivo mayor que 2). En la CC estricta, se cuenta uno para cada || o && en el código fuen-

te de la condición asociada a un vértice de decisión (aquel con semigrado positivo mayor que 1). La CC esencial mide la cantidad de código desestructurado en una función. Para esto, G se reduce contrayendo los vértices y arcos que representan primitivas de programación estructuradas (se forman aglomerados); se procede desde el nivel más profundo de anidamiento hacia el exterior, hasta que no se puede reducir más el grafo. Después se calcula CC para el grafo reducido.

Software: CodeAnalyzer

Métricas:

AvgLineLength longitud de línea media
CodeToCommWhite cociente del número de líneas de código fuente, y la suma del número de líneas con comentarios y el número de líneas con un espacio en blanco
CodeToWhite cociente del número de líneas de código fuente y el número de líneas con un espacio en blanco
CodeToTotal cociente del número de líneas de código fuente y el número de líneas totales

Software: RSM 6.52

Métricas:

NumIf número de palabras clave `if` en el código
NumElse número de palabras clave `else`
NumSwitch número de palabras clave `switch`
NumCase número de palabras clave `case`
NumWhile número de palabras clave `while`
NumDo número de palabras clave `do`
NumFor número de palabras clave `for`
NumBreak número de palabras clave `break`
NumReturn número de palabras clave `return`
NumGoto número de palabras clave `goto`
NumConst número de palabras clave `const`
NumEnum número de palabras clave `enum`
NumDefault número de palabras clave `default`
NumString número de cadenas de literales
NumPar número de paréntesis
NumBrace número de llaves
NumBracket número de corchetes

Software: Metre 2.3

Métricas:

MaxDepth profundidad máxima de una estructura de control

Software: generador de casos de prueba basado en EDAs

Métricas:

NumBranch número de ramas en el código fuente

Los valores de esta métrica se obtuvieron utilizando nuestro generador de casos de prueba.

Referencias

[1] T.J. McCabe, A Complexity Measure, *IEEE Transactions on Software Engineering* 12(2):208–220, 1976.

[2] W.H. Press, B.P. Flannery, S.A. Teukolsky, W.T. Vetterling, *Numerical Recipes in C. The Art of Scientific Computing*, Cambridge University Press, Cambridge, 1988.

[3] R. Sagarna, J.A. Lozano, On the performance of Estimation of Distribution Algorithms applied to Software Testing, *Applied Artificial Intelligence* 19(5):457–489, 2005.

[4] R. Sagarna, J.A. Lozano, Métricas de Software para Predecir el Rendimiento de los Algoritmos de Estimación de Distribuciones en la Generación de Casos de Prueba, enviado para publicación.