МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Запорізький національний технічний університет

Кафедра програмних засобів

РЕФЕРАТ

**Метрики складності управління даними.**

**Управління даними і потоками даних.**

з дисципліни

«Якість програмного забезпечення та тестування»

Виконала:

студентка групи КНТ-136сп О.С. Симоненко

2018

## Зміст

[Вступ 3](#_Toc527371629)

[Метрики складності потоку даних 4](#_Toc527371630)

[Висновок 8](#_Toc527371631)

[Список літератури 9](#_Toc527371632)

# Вступ

При оцінці складності програм, як правило, виділяють три основні групи метрик: метрики розміру програм, метрики складності потоку керування програм і метрики складності потоку даних програм.

Оцінки першої групи найбільш прості і тому набули широкого поширення. Традиційною характеристикою розміру програм є кількість рядків вихідного тексту. Під рядком розуміється будь-який оператор програми.

Безпосереднє вимірювання розміру програми, незважаючи на свою простоту, дає хороші результати. Оцінка розміру програми недостатня для прийняття рішення про її складності. Але цілком може бути застосована для класифікації програм, істотно розрізняються обсягами. При зменшенні відмінностей в обсязі програм на перший план висуваються оцінки інших факторів, що впливають на складність. Таким чином, оцінка розміру програми є оцінка за номінальною шкалою, на основі якої визначаються тільки категорії програм без уточнення оцінки для кожної категорії.

Друга найбільш представницька група оцінок складності програм - метрики складності потоку керування програм. Як правило, за допомогою цих оцінок оперують або щільністю керуючих переходів усередині програм, або взаємозв'язками цих переходів.

Інша група метрик складності програм - метрика складності потоку даних, тобто використання, конфігурації і розміщення даних в програмах.

# Метрики складності потоку даних

На відміну від метрик складності управління, метрики складності потоку даних в більшій мірі орієнтовані на оцінку вихідного тексту програм на мовах високого рівня, відповідно, з високорівневими типами даних. Модифікація більшості наведених нижче метрик для низькорівневого уявлення, тобто для бінарного коду, як мінімум вимагає деякого аналізу бінарного коду для відновлення даних.

Однією з метрик складності потоку даних програм є метрика Чепіна. Існує кілька її модифікацій. Розглянемо більш простий, а з точки зору практичного використання - досить ефективний варіант цієї метрики.

   Суть методу полягає в оцінці інформаційної міцності окремо взятого програмного модуля за допомогою аналізу характеру використання змінних зі списку введення-виведення.

   Вся безліч змінних, що складають список введення-виведення розбивається на чотири функціональні групи

1. Р - вводяться змінні для розрахунків і для забезпечення виведення. Прикладом може служити використовувана в програмах лексичного аналізатора змінна, що містить рядок вихідного тексту програми, тобто сама змінна не модифікується, а лише містить вихідну інформацію.

2. М - модифікуються або створювані всередині програми змінні.

3. C - змінні, що беруть участь в управлінні роботою програмного модуля (керуючі змінні).

4. Не використовувані в програмі ( "паразитні") змінні. Оскільки кожна змінна може виконувати одночасно кілька функцій, необхідно враховувати її в кожній відповідній функціональній групі.

Далі вводиться значення метрики Чепіна:

Q = a1P + a2M + a3C + a4T,

де a1, a2, a3, a4 - вагові коефіцієнти.

Вагові коефіцієнти використані для відображення різного впливу на складність програми кожної функціональної групи. На думку автора метрики, найбільшу вагу, що дорівнює трьом, має функціональна група С, так як вона впливає на потік управління програми. Вагові коефіцієнти інших груп розподіляються наступним чином: a1 = 1; a2 = 2; a4 = 0.5. Ваговий коефіцієнт групи T не дорівнює нулю, оскільки "паразитні" змінні не збільшують складності потоку даних програми, але іноді ускладнюють її розуміння. З урахуванням вагових коефіцієнтів вираз прийме вид:

Q = P + 2M + 3C + 0.5T.

Ще одна метрика складності потоку даних - Спен. Визначення Спен грунтується на локалізації звернення до даних усередині кожної програмної секції. Спен - це число тверджень, які містять даний ідентифікатор, між його першою і останньою появою в тексті програми. Ідентифікатор, що з'явився n раз, має Спен, рівний n-1.

Спен визначає кількість контролюючих тверджень, що вводяться в тіло програми при побудові траси програми по цьому ідентифікатору в процесі тестування і налагодження.

Метрика звернень до глобальних змінних. В залежно від наявності в програмі реальних звернень до змінної, формуються фактичні і можливі пари (модуль, глобальна змінна), які говорять про те, скільки разів модуль дійсно отримував доступ до глобальних змінним, і скільки разів він міг цей доступ отримати. Ставлення фактичного числа звернень до можливого показує ймовірність несанкціонованого зміни глобальної змінної.

Метрика Кафура (Henry & Kafura) будується на основі концепції інформаційних потоків (зустрічається також під назвою "Fan in / out complexity"). Ця міра оперує локальними і глобальними потоками інформації і вводить оцінку інформаційної складності процедури і модуля (через потоки або щодо деякої структури даних). Інформаційна складність враховує також складність тексту, вимірювану через одну з метрик обсягу, наприклад, Холстеда або LOC; може враховуватися також цикломатична складність.

Вводяться поняття локального і глобального потоку: локальний потік інформації з A в B існує, якщо:

* Модуль А викликає модуль В (прямий локальний потік);
* Модуль В викликає модуль А і А повертає В значення, яке використовується в В (непрямий локальний потік);
* Модуль С викликає модулі А, В і передає результат виконання модуля А в В;
* Глобальний потік інформації з А в В через глобальну структуру даних D існує, якщо модуль А поміщає інформацію в D, а модуль В використовує інформацію з D.

На основі цих понять вводиться величина I - інформаційна складність процедури:

I = length \* (fan\_in \* fan\_out) ^ 2

В ній:

length - складність тексту процедури (міряється через котрусь із метрик обсягу, типу метрик Холстеда, Маккейб, LOC і т.п.);

fan\_in - число локальних потоків всередину процедури плюс число структур даних, з яких процедура бере інформацію;

fan\_out - число локальних потоків з процедури плюс число структур даних, які оновлюються процедурою;

Можна визначити інформаційну складність модуля як суму інформаційних складнощів входять до нього процедур.

Наступний крок - розглянути інформаційну складність модуля щодо деякої структури даних. Інформаційна міра складності модуля щодо структури даних:

J = W \* R + W \* WrRd + WrRd x R + WrRd \* (WrRd - 1)

В формулі:

W - число процедур, які тільки оновлюють структуру даних;

R - Тільки читають інформацію зі структури даних;

WrRd - і читають, і оновлюють інформацію в структурі даних.

Метрика, заснована на обліку потоку даних.

Також одна з метрик інформаційна складність модуля щодо структури даних обчислюється на основі числа процедур, які тільки оновлюють, тільки читають, або і оновлюють, і читають деяку структуру даних.

Міра Ов'єдо (Oviedo). Програма розбивається на лінійні непересічні ділянки - так звані промені операторів, які утворюють граф потоку керування програми. Міра складності кожного променя визначається, як сума кількостей визначальних входжень для кожної використовуваної в промені змінної. Автор метрики припускає, що знайти відношення між визначеннями і використаннями змінної всередині променя простіше, ніж між променями, і що число різних визначальних входжень в кожному промені важливіше, ніж загальне число використовують входжень змінної в кожному промені. Зустрічається також інше визначення міри Ов'єдо:

C = aCF + bDF,

де CF - складність потоку управління, що враховує тільки число дуг графа;

 DF - складність потоку даних, що обчислюється як сума складнощів потоків даних базових блоків програми, причому складність потоку даних блоку є число змінних, які використовуються, але не визначаються в блоці;

 a, b - вагові коефіцієнти, які можуть бути прийняті рівними 1.

# Висновок

У чистому вигляді всі перераховані метрики недостатні для того, щоб охарактеризувати складність аналізу трас програм в бінарних кодах. Застосування метрик складності потоку даних до двійкового коду ускладнюється також тим, що компілятор може використовувати для розміщення змінних як пам'ять, так і регістри процесора, і потрібно додатковий аналіз з метою зведення змінних в єдиний список. Метрика Чепіна може представляти інтерес головним чином при створенні комбінованих і гібридних метрик складності заплутаного коду, особливо, що використовує непрозорі. Метрика Спен легко обчислювана

і може бути застосована до трас програм. Метрики звернень до глобальних змінних і Кафура видаються занадто складними для використання при аналізі трас. Метрика складності модуля щодо структури даних не може бути застосована для двійкових кодів, так як інформація про складні структурах даних втрачена при компіляції. Міра Ов'єдо може розглядатися, як перехідна від «чистих» метрик складності потоку даних до комбінованих, так як враховує потік управління і зростає з числом «міжблочних посилань», коли змінна визначається

і використовується в різних базових блоках програми.

# Список літератури

1. Інженерія якості програмного забезпечення:навчальний посібник / Г.В Табунщик, Р.К. Кудерметов, Т.І. Брагіна. – Запоріжжя: ЗНТУ, 2013. – 180 с.
2. <http://www.ispras.ru/preprints/docs/prep_25_2013.pdf>
3. <http://www.csom.ru/category/articles/metriki_koda_i_ih_prakticheskaya_realizaciya_v_subversion_i_clearcase_chast_1_-_metriki.html>
4. <http://www.met-rix.narod.ru/page2.htm>
5. <http://www.ict.edu.ru/ft/005437/nwpi263.pdf>
6. <https://studopedia.info/1-35619.html>
7. https://studopedia.su/14\_50413\_metriki-slozhnosti-potoka-dannih-programm.html