Міністерство освіти і науки України
Департамент науки і освіти Харківської облдержадміністрації
Харківське територіальне відділення МАН України

Відділення: комп'ютерних наук
Секція: комп'ютерні системи та мережі

РОЗРОБКА МЕТОДА ПОШУКУ ТА УСУНЕННЯ ПОВТОРЮВАНИХ ЧАСТИН У ПОЧАТКОВОМУ КОДІ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Роботу виконав:
Човпан Ігор Сергійович,
учень 11 класу Харківського Навчально-виховного комплексу №45 «Академічна гімназія» Харківської міської ради Харківської області

Науковий керівник:
Руккас Кирило Маркович,
професор кафедри теоретичної та прикладної інформатики механіко-математичного факультету Харківського національного університету ім. В.Н. Каразіна, доктор технічних наук, доцент

Харків – 2020

Тези

...

ЗМІСТ

ВСТУП

Ваш ВСТУП

РОЗДІЛ 1.
ХАРАКТЕРИСТИКА ІСНУЮЧИХ МЕТОДІВ ЗНАХОДЖЕННЯ ДУБЛІКАТІВ У КОДІ

1.1. Визначення основних видів повторюваних частин

Як зазначено у роботах [CH79.pdf],[SUB153536.pdf],[astclones-wcre06.pdf], виділяється 4 головних типи повторюваних частин.

- I тип – повна копія без модифікацій, окрім пробілів та коментарів;
- II тип – синтаксично однакова копія, змінюються лише назви змінних, назв функцій, тощо;
- III тип – копія з подальшими змінами; доданими, зміненими або видаленими інструкціями;
- IV тип – частина, що робить ідентичні обчислювання; синтаксично імплементована інакше, ніж інша частина.

1.2. Основні методи пошуку повторюваних частин

Існує багато прийомів, що використовуються для пошуку повторюваних частин у початковому коді програмного забезпечення.

Перелічим основні методи пошуку:

- пошук збігу рядків початкового коду;
- використання токенів;
- метод порівняння функцій;
- застосування графа програмних залежностей;
- метод порівняння дерев.

Далі визначимо усі переваги і недоліки кожного з методів.

1.2.1 Пошук збігу рядків початкового коду

Обчислюється ступінь схожості для кожної пари рядків за допомогою відстані Левенштейна. Емпірично встановлюється мінімальна величина, за якої вважається, що 2 рядки є копіями одна одну.

Переваги цього метода:

- добре знаходить копії I типу;
- невеликий час виконання порівняно з іншими методами;
- підтримка будь-якої мови програмування.

Недоліки метода:

- велика кількість хибнонегативних результатів;
- нестійкість до різних ”шумів”: коментарів, змінених назв функцій або змінних, тобто неможливість знайти дублікати II та III типу.
- Не враховуються особливості мови програмування.

Прикладом використання є програма PMD.

1.2.2 Використання токенів

Метод використовує токенізатор. У ньому початковий код розбивається на токени, при пошуці порівнюються послідовності токенів. Головною перевагою цього методу є стійкість до переформатування початкового коду, зміні назв змінних. Недоліком є те, що токенізатори враховують тільки базові особливості мови програмування. Через те, що токени усе одно порівнюються як рядки, присутня досить велика похибка при розпізнаванні копій. Наприклад, наступна пара токенів може вважатися як копії один одному:

```
reallySoLongNameThatYouCouldNotStandIt1
```

та

```
reallySoLongNameThatYouCouldNotStandIt2.
```

Прикладом використання є програма CCFnderX.

1.2.3 Метод порівняння функцій

За допомогою парсера мови програмування знаходять усі функції у початковому коді. Далі усі ці функції порівнюються між собою або за допомогою спеціально обраної ”поганої”геш-функції, або за допомогою обчислення коефіцієнту схожості (наприклад, коеф. Жаккара). Метод гарно знаходить збіги між різними функціями, розпізнаються копії I-III типу, проте він не може знайти повторювани частини всередині коду. Прикладом використання є [https://ieeexplore.ieee.org/documen

1.2.4 Застосування графа програмних залежностей

Згідно з [The_Program_Dependence_Graph..pdf] Граф програмних залежностей (далі просто граф) – представлення програми як графа, у якому кожна вершина - інструкція у програмі, а також зв’язані з цією інструкцією оператори та операнди; ребрами у такому графі є дані, від яких залежить виконання цієї інструкції та умови, за яких ця інструкція виконається. Дві частини програми вважаються ідентичними, якщо їх графи ізоморфні. Головною перевагою є те, що цей граф не залежить від переставлення інструкції програми; зміни назв функцій, змінних, тощо; не залежить від аспектів реалізації. Недоліки методу:

- Дуже довгий час роботи, оскільки завдання пошуку ізоморфних підграфів є NP-повною, і може бути вирішена за поліноміальний час тільки для планарних графів, що не обов’язково виконається для графа програмних залежностей;
- Такий метод не зможе знайти дублікати у коді, який не виконується у загальному випадку, оскільки у граф додаються лише виконані інструкції. Приклад використання: [kdd06_gplag.pdf].

1.2.5 Метод порівняння дерев

РОЗДІЛ 2.
ЗАПРОПОНОВАНИЙ МЕТОД

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

Ваш список источников