Одиннадцатая независимая научно-практическая конференция «Разработка ПО 2015»

22 - 24 октября, Москва

Обнаружение клонов в ПО: от тяжеловесных алгоритмов к настольному инструменты программиста

Сухинин Александр

СПбПУ/JetBrains

Ахин Марат

СПбПУ/JetBrains

Программные клоны

- Клоны – экземпляры похожих или одинаковых фрагментов исходного кода.

Почему это плохо?

- Ухудшают качество кода
- Повышают стоимость его сопровождения и развития
- Приводят к сложностям понимания системы
- Приводят к появлению новых и тиражированию старых ошибок

Типы клонов. Тип I.

Без учета форматирования и комментариев

```
if (a >= b) {
    c = d + b; //Comment1
    d = d + 1;}
else
    c = d - a; //Comment2
```

```
if (a>=b)
{ // Comment1’’
    c=d+b;
    d=d+1;
}
else // Comment2’’
    c=d-a;
```

Типы клонов. Тип II.

Без учета переименований

```
if (a >= b) {
    c = d + b; //Comment1
    d = d + 1;}
else
    c = d - a; //Comment2
```

```
if (m >= n)
{ // Comment1
    y = x + n;
    x = x + 5; //Comment3
}
else
    y = x - m; //Comment2
```

Типы клонов. Тип III.

Без учета добавлений и удалений

```
if (a >= b) {
    c = d + b; //Comment1
    d = d + 1;}
else
    c = d - a; //Comment2
```

```
if (a >= b) {
    c = d + b; //Comment1
    e = 1; // This statement is added
    d = d + 1; }
else
    c = d - a; //Comment2
```

Типы клонов. Тип IV.

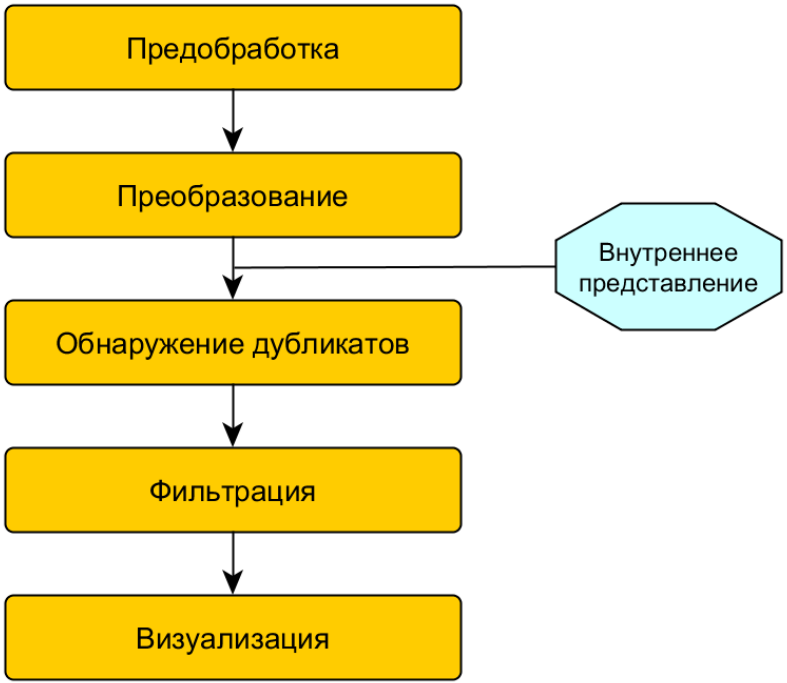
Семантически похожие фрагменты

```
int i, j=1;
for (i=1; i<=VALUE; i++)
    j=j*i;
```

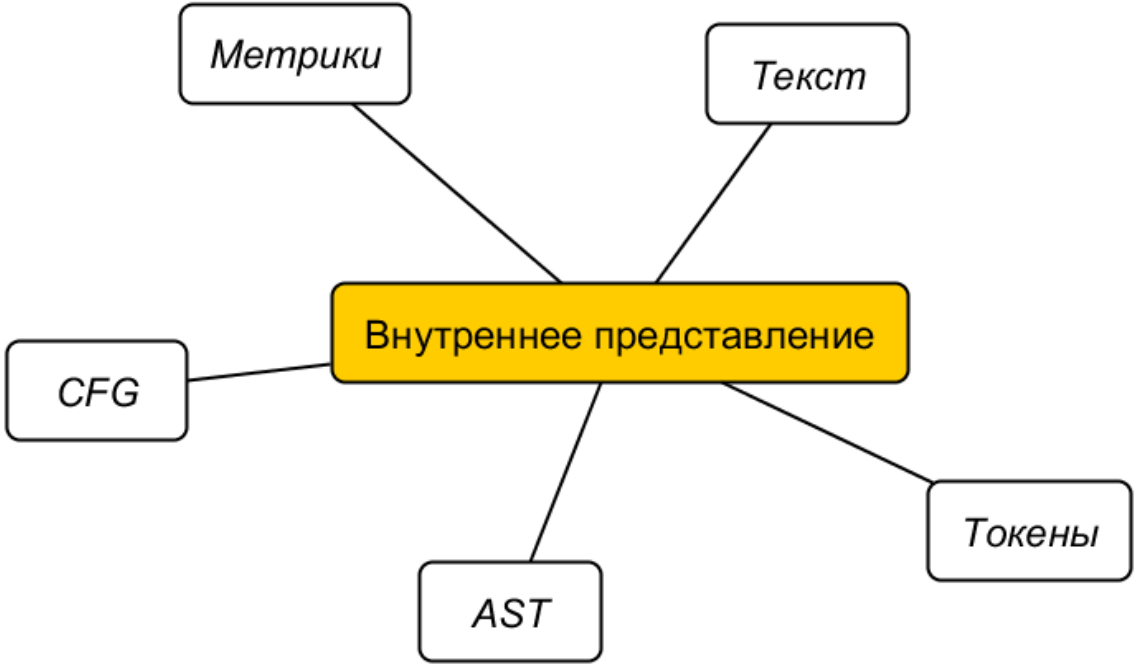
```
int factorial(int n) {
    if (n == 0) return 1;
    else return n * factorial(n-1);
}
```

Этапы

Использование промежуточной структуры позволяет применить более эффективные и сложные методы.



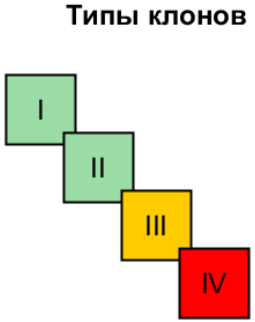
Известные подходы



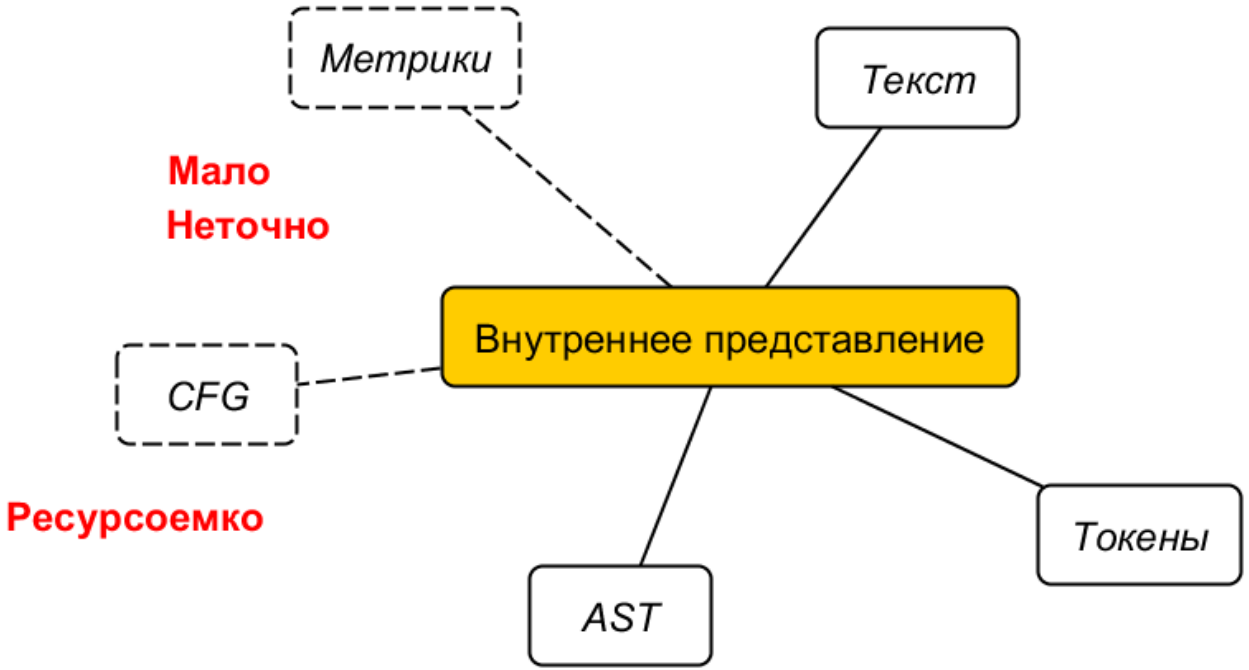
А что мы хотим?

- ~~Быстро! Дешево! Качественно!~~

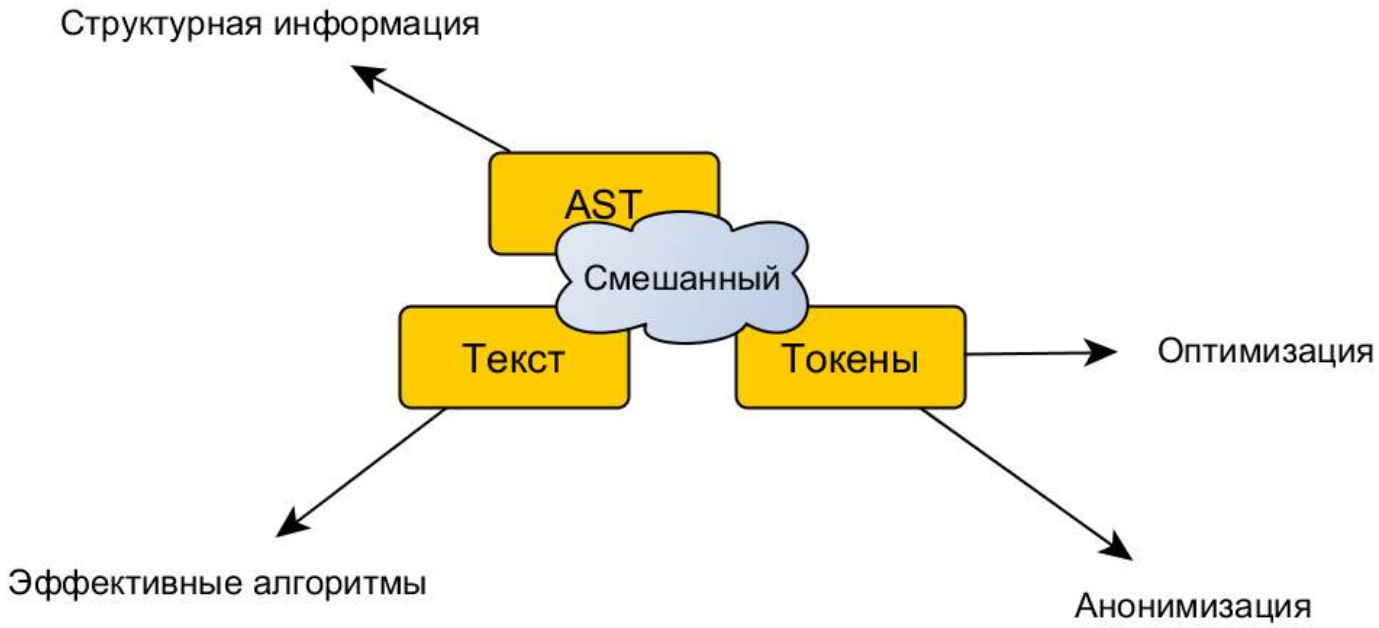
- Быстрый анализ
- Точность



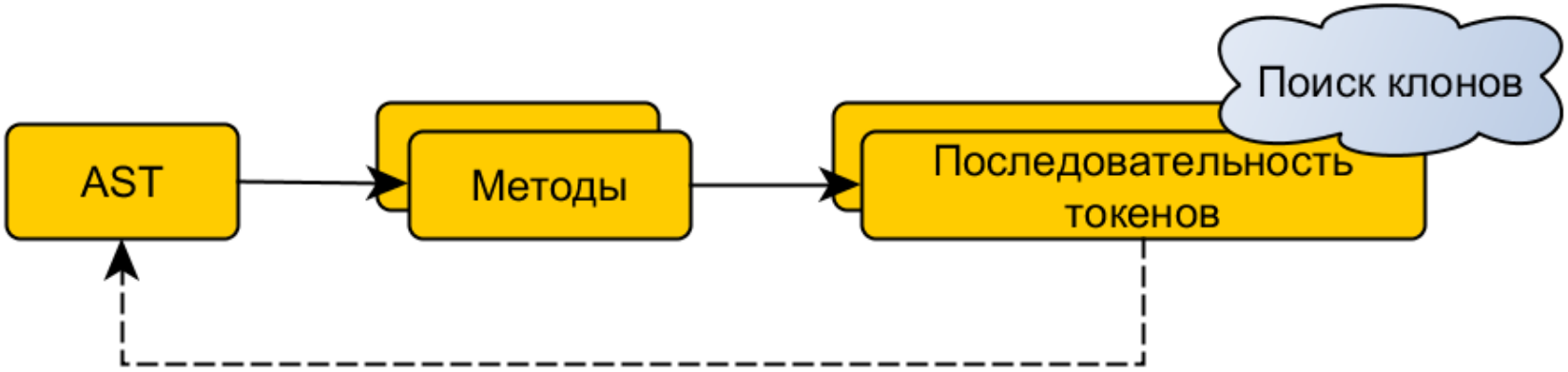
Известные подходы



Используемый подход



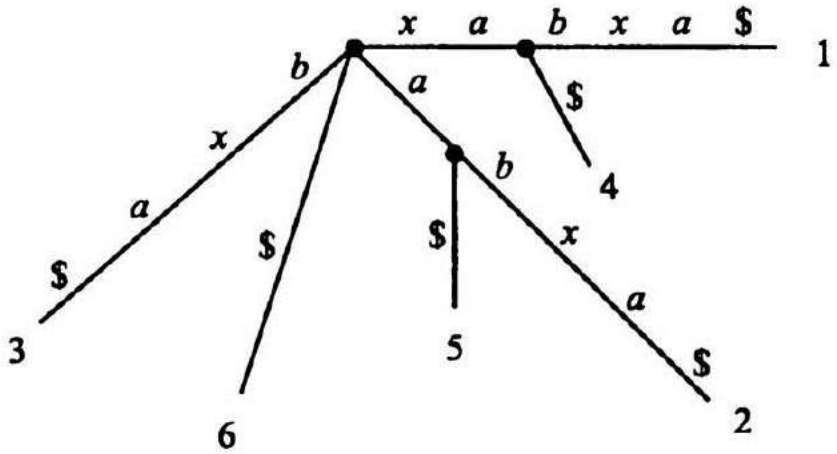
Используемый подход



Суффиксное дерево

Дерево содержащее все суффиксы некоторой строки.

Пример: xabxa$



Суффиксное дерево

- Входит ли строка S_m в строку S_n?

- O(n) времени на построение.
- O(n) используемой памяти.
- O(m) времени на проверку.

Суффиксное дерево

- Можно добавить строку S_m .
- Можно удалить строку S_m .

- За O(m)

Суффиксное дерево

- Можно добавить строку S_m .
- Можно удалить строку S_m .

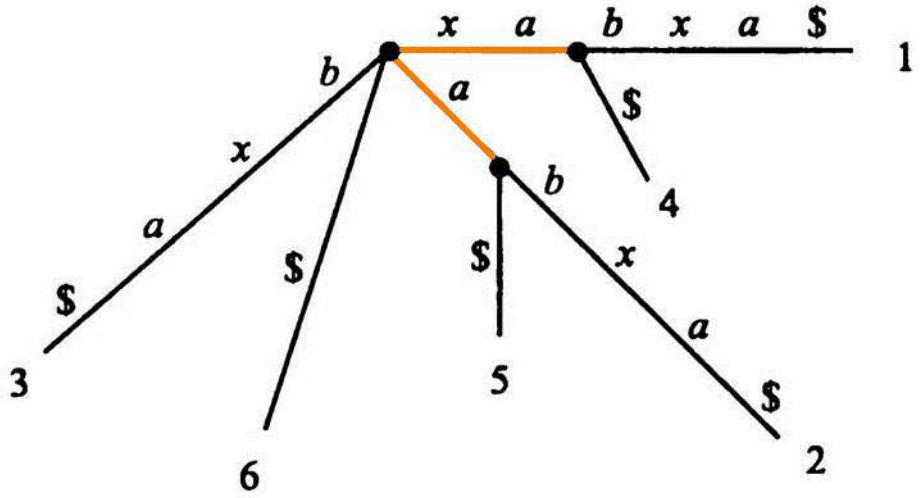
Обновить фрагмент

- За O(m)

Суффиксное дерево

- Не только текст
- Добавление и удаление
- Качество
- Java

А где клоны?



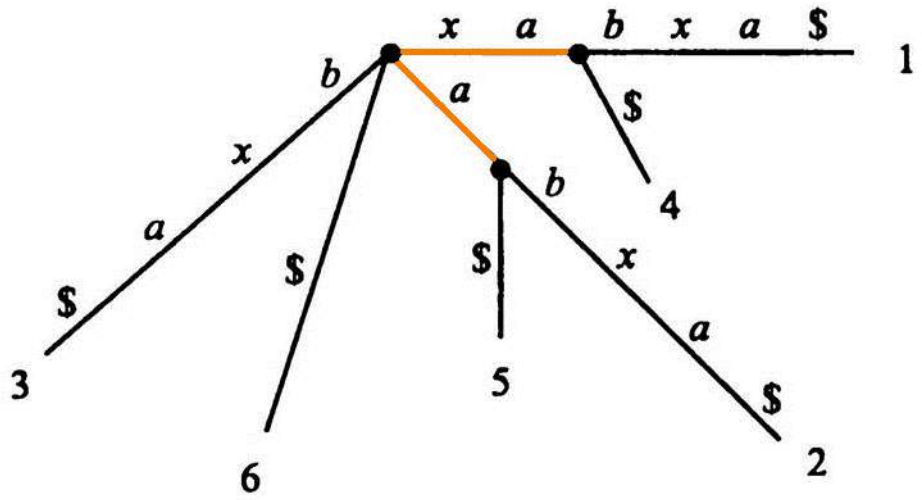
Почему?

- Путь – подстрока.
- Развилка – несколько путей с одинаковым началом.

- Разные подстроки с одинаковой частью?

Извлечение

Ленивый обход потомков каждого узла.

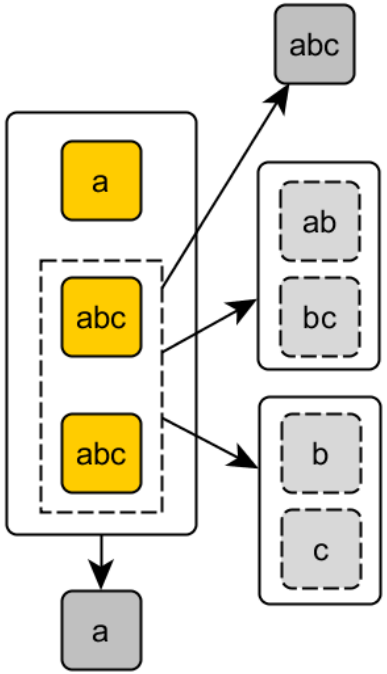


Фильтрация

- Короткие клоны – неинтересно

Фильтрация

Чем длиннее исходный клон, тем больше побочных результатов.



Когда фильтровать?

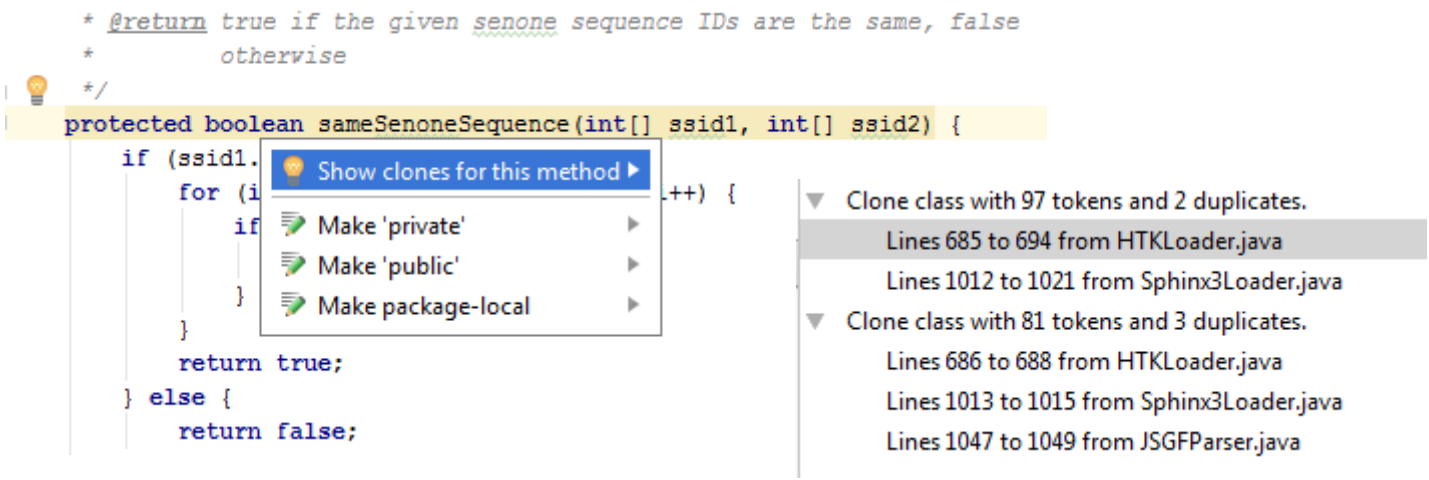
- Часть другого набора клонов
- Нет новых дубликатов

Как фильтровать?

- Часть другого набора клонов
  - суффиксные ссылки?
- Нет новых дубликатов
  - такое же количество?

- Очень быстро!

Plugin


* @return true if the given senone sequence IDs are the same, false
* otherwise
*/
protected boolean sameSenoneSequence(int[] ssid1, int[] ssid2) {
if (ssid1.
for (i
if
}
}
return true;
} else {
return false;
Show clones for this method
Make 'private'
Make 'public'
Make package-local
++) {
Clone class with 97 tokens and 2 duplicates.
Lines 685 to 694 from HTKLoader.java
Lines 1012 to 1021 from Sphinx3Loader.java
Clone class with 81 tokens and 3 duplicates.
Lines 686 to 688 from HTKLoader.java
Lines 1013 to 1015 from Sphinx3Loader.java
Lines 1047 to 1049 from JSGFParser.java

Plugin. Сравнение.

- Библиотека sphinx4-5prealpha
  - Около 100 тыс. LOC

- PC
  - Intel Core i5 760
  - 8 Гб

Сравнение

	Прототип	Duplicates tool (IDEA)	On-the-fly detection (IDEA)
Типы клонов	I, II	I, II	I, II
Обнаружено классов	2903	843	≪ 843
Время полного анализа	6 сек	121 сек	?
Инкрементальный анализ	Доли секунд	Нет	Доли секунд
Анонимизация	Полная	Настраиваемая	Слабая

Что дальше?

- GUI
- Фильтрация

Что дальше?

- GUI
- Фильтрация
- Рефакторинг
- Клоны III типа

Ссылки

- Библиотека для работы с суффиксным деревом
  - https://github.com/suhininalex/SuffixTree
- Прототип плагина для IntelliJ IDEA
  - https://github.com/suhininalex/IdeaClonePlugin