

Форматирование по эталону

Подкопаев Антон, podkoav239@gmail.com

JetBrains, СПбГУ

Руководитель: к.ф.-м.н. Булычев Д.Ю.

27 сентября 2013

Печать AST



Область применения

Языковые процессоры

- Синтаксический анализ
- Преобразование
- Представление результата
 - Код программы
 - ...

Печать AST с помощью шаблонов

- Модельный язык L
- Специальные шаблоны
- Расширенный парсер

```
t_start  
write(@-)  
t_end
```

```
t_start  
write(@|  
      @|  
)  
t_end
```

Печать AST с помощью шаблонов. Алгоритм

- Сопоставление деревьев сверху-вниз
- $O(T \times B \times M)$
 - T — размер дерева
 - B — максимальный размер образца
 - M — количество образцов

Постановка задачи

- Апробирование для Java
- Плагин для IDEA
- Алгоритм, перебирающий все подходящие шаблоны

Постановка задачи

- Апробирование для Java
- Плагин для IDEA
- Алгоритм, перебирающий все подходящие шаблоны
- Отказ от специализированного парсера
- Получение шаблонов из обычных исходников

```
#define true false
//happy codings :)
```

Эталон

```

                = (
                    255.
                    lambda
                        V
                        ,B,c
                    :c and Y(V*V+B,B,c
                        -1)if(abs(V)<6)else
                        2+c-4*abs(V)**-0.4)/\
                    x=1500,1000;C=range(v*x
                    );import struct;P=struct.pack('B,\
j = '<QIIHHHH'\,open('M.bmp','wb')).write
for X in j('BM'+P(M,v*x*3+26,26,12,v,x,1,24))or C:
    Y=_;j(P('BBB'),*(lambda T:(T*80+T**9
        *i-950*T **99,T*70-880*T**18+701*
        T **9,T*i**(1-T**45*2)))(sum(
            Y(0,(A%3/3.+X%v+(X/V+
                A/3/3.-X/2)/11)*2.5
                /X
                -2.7,1)**2 for \
                    A
                    in C
                    [:9]))
                    /9)
                )

```

Для переформатирования


```
#define true false
//happy codings :)
```

Эталон

```

                                = (
                                255.
                                lambda
                                B,c
                                :c and Y(V*V+B,B,c
                                -1)if(abs(V)<6)else
                                2+c-4*abs(V)**-0.4)/1
                                x=1500,1000;Carange(v*x
                                );import struct;P=struct.pack('f');
                                j = '<QIIHHHHH'.open('M.bmp','wb').write
                                for X in j('BM'+P(M,v*x*3+26,26,12,v,x,1,24))or C:
                                Y=_;j(P('BBB'+(lambda T:(T*80+T**9
                                *1.950*T **99,T*70-880*T**18+701*
                                T**9,T**45*(1-T**45*2)))(sum(
                                Y(0,(A%3/3.+X%v+(X/v+
                                A/3/-X/2)/1))**2.5
                                /X
                                -2.7,1)**2 for
                                A
                                in C
                                [:9]))
                                /9)
                                )

```

Для переформатирования

```
int getRandomNumber()
{
    return 4; // chosen by fair dice roll.
              // guaranteed to be random.
}
```

Результат

Трудности

- Вырезка шаблонов
- Комментарии
- Экспонента в списках и бинарных выражениях
 - Фиксированные варианты
- Требуется большой эталон
- Несколько стилей в эталоне
- Экспоненциальность алгоритма
 - Локальный выбор

В будущем

- Нормализация шаблонов
- Анализ эталона на полноту
- Сохранение пользовательского форматирования?
- Интеграция с Formatter?
- Машинное обучение?

Результат

- Прототип плагина
bitbucket.org/anlun/printerplugin
- Расширен общий подход

Можно ли проще?

Вообще-то да...

```
class Country{
private:
    char* name;
    int religion, system;
    float area, population, budget, military;
    bool sea;

public:
    Country(void); //конструктор по умолчанию
    Country(char*, int, int, float, float, float, float, bool); //конструктор с параметрами
    Country(Country*); //конструктор копии
    ~Country(void); //деструктор

    char* getName();
    int getReligion();
    int getSystem();
    float getArea();
    float getPopulation();
    float getBudget();
    float getMilitary();
    bool isSea();
    void setArea(float);
    void setName(char*);
    void setReligion(int);
    void setSystem(int);
    void setPopulation(float);
    void setBudget(float);
    void setMilitary(float);
    void setSea(bool);
};
```



```
Before Parentheses
  [ ] Method declaration parentheses
  [ ] Method call parentheses
  [x] 'if' parentheses
  [x] 'for' parentheses
  [x] 'while' parentheses
  [x] 'switch' parentheses
  [x] 'try' parentheses
  [x] 'catch' parentheses
  [x] 'synchronized' parentheses
  [ ] Annotation parameters

Around Operators
  [x] Assignment operators (=, +=, ...)
  [x] Logical operators (&&, ||)
  [x] Equality operators (==, !=)
  [x] Relational operators (<, >, <=, >=)
  [x] Bitwise operators (&, |, ^)
  [x] Additive operators (+, -)

@Annotation(para1 = "value", para2
@SuppressWarnings({"ALL"})
public class Foo<T, U> {
    int[] X = new int[] {1, 3, 5, 6, 7,
    public void foo(int x, int y) {
        Runnable r = () -> {
            ...
        };
        Runnable r1 = this::bar;
        for (int i = 0; i < x; i++) {
            y += (y ^ 0x123) << 2;
            ...
        }
        do {
            try (MyResource r1 = getResour
            if (0 < x && x < 10) {
                while (x != y) {
                    x = f(x * 3 + 5);
                }
            } else {
                synchronized (this) {
                    switch (e.getCode()) {
```