

Прототип инструмента для получения и анализа графа потока управления открытых исходных текстов С/С++ на основе универсального промежуточного представления

Пустыгин А. Н.
Ковалевский А. А.
Огуречникова Е. А.
Субботин А. А.

Цели написания прототипа

1. Проба существующих поддерживаемых инструментов извлечения из компилятора C/C++ внутренней модели исходного текста- дерева разбора
- 2 Проверка введенных промежуточного и эквивалентных представлений при анализе существующих проектов
- 3 Оценка степени сложности использования предложенных эквивалентных представлений для статического анализа
- 4 Проба графической интерпретации эквивалентных представлений для отображения результатов обработки эквивалентных представлений

Известные ближайшие аналоги

Прототип XBLiteGen – написан для построения блок-схем алгоритмов на языке-клоне BASIC Получено свидетельство о госрегистрации

Частичные решения:

- Elsa, Bison, Flawfinder, GCC-XML
- scrML, JavaML, c2xml, C++2XML
- Dehydra, Treegydra
- Clang

Выбранные условия разработки

- ОС Ubuntu 10.04
- Язык C/C++
- Компилятор clang
<http://clang.llvm.org/>
- Библиотека libclang 3.2,
из проекта Clang

Общий алгоритм функционирования прототипа

- Обход дерева разбора компилятора производится через интерфейс «посетитель», предоставляемый библиотекой clang (фронт-энд компилятора) Узлы AST представлены в виде специальных атрибутированных объектов — курсоров.

Для получения того или иного атрибута курсора, библиотека предоставляет набор функций

Получение срезов диаграммы путем фильтрации по условию с помощью стандартного xml-парсера

Графический Интерфейс прототипа - web-браузер

JavaScript код обеспечивает интерактивность эквивалентного представления (обратную связь с исходным текстом, другими представлениями, задание параметров срезов и их применение, навигация по странице).

Реализованные функции

- Построение эквивалентного представления в виде диаграммы потока управления в **HTML**-формате
- Возможность получения срезов диаграммы по ряду условий
- Переход в исходный текст из любого блока диаграммы через любой текстовый редактор операционной системы
- Возможность связи эквивалентного представления в виде диаграммы потока с другими эквивалентными представлениями

Атрибуты построения срезов

- имя идентификатора,
- метод доступа (любой, чтение, запись, вызов),
- тип идентификатора (любой, переменная, поле, функция, метод),
- модификатор доступа (любой, public, private, protected),
- имя родительского класса и/или области видимости,
- специальные параметры фильтрации (рекурсивные вызовы, инициализация «мусором», присваивание в условиях, цикл while/do-while, цикл for)

Исходный текст теста на «полноту»

```
#include <stdio.h>
#include <stdlib.h>
#include "tools.h"

using namespace tools;

long fact(long n)
{
    return n > 0 ? n * fact(n - 1) : 1;
}

int main(int argc, char *argv[])
{
    int tool_ids[]={1,2,3};
    Tools *tools=new Tools(tool_ids,sizeof(tool_ids)/sizeof(tool_ids[0]));
    Tool* tool=tools->getTool(argc>1?atoi(argv[1]):1);
    if(!tool)goto error;
    tool->setData((void*)"Test data");
    return 0;
error:
    puts("Failed to get tool by ID");
    return -1;
}
```

Результаты теста на «производительность»

Входные файлы: `sxx2xml.cpp`, `utils.h`, `Index.h` (42 КБ)

Выходные файлы: `sxx2xml.xml` (350 КБ), `sxx2xml.html` (2.1 МБ)

Время генерации файла промежуточного представления
`sxx2xml.xml`: 350 КБ за 350 мс

Время генерации файла эквивалентного
представления `sxx2xml.html`: 2.1 Мб за 475 мс

Результаты опытной разработки

- Опробован формат универсального промежуточного представления исходных текстов программ для языка C/C++
- Опробовано применения стандартного фронт-энда компилятора для извлечения дерева разбора
- формат универсального промежуточного представления исходных текстов программ для языка C/C++ использован для получения эквивалентных представлений путем визуализации средствами стандартного web-браузера

Файл промежуточного текстового представления в XML-формате

```
1 long fact(long n)
2 {
3     return n > 0 ? n * fact(n - 1) : 1;
4 }
```

Вычисление факториала

```
<files>
  <file name="/home/mozy/tests/test/stest.cc" language="C/C++">
    <function name="fact" rtype="long" argc="1" line="1" col="6" line2="4" col2="2" file="/home/mo:
      <argument type="long" name="n" line="1" col="16"/>
      <block line="2" col="1" line2="4" col2="2">
        <return line="3" col="4">
          <terop line="3" col="11">
            <binop type=">" line="3" col="11">
              <var name="n" type="long" line="3" col="11"/>
              <literal value="0" type="int" line="3" col="15"/>
            </binop>
            <binop type="*" line="3" col="19">
              <var name="n" type="long" line="3" col="19"/>
              <call name="fact" rtype="long" line="3" col="23" line2="3" col2="34">
                <var name="fact" type="functionproto" line="3" col="23"/>
                <binop type="-" line="3" col="28">
                  <var name="n" type="long" line="3" col="28"/>
                  <literal value="1" type="int" line="3" col="32"/>
                </binop>
              </call>
            </binop>
            <literal value="1" type="int" line="3" col="37"/>
          </terop>
        </return>
      </block>
    </function>
```

Эквивалентное представления исходного текста в виде диаграммы потока управления в окне web-браузера

Начальное состояние диаграммы в окне браузера

The screenshot shows a web browser window with a single tab titled "test.html". The address bar displays the file path: `file:///home/mozy/tests/test.html`. The main content area shows a code editor with the following code:

```
File: tools.h
3 namespace tools { }
File: test.cc
7 long fact ( long n )
12 int main ( int argc, char *argv[] )
```

Below the code editor, there is a control flow diagram interface. It includes a list of filters with checkboxes:

- recursion (рекурсия)
- trash init (инициализация мусором)
- assign in condition (= в условии)
- while цикл
- for цикл

At the bottom of the interface, there are input fields for "ID:" and "PID:", dropdown menus for "Тип:" (set to "Любой") and "Доступ:" (set to "Любой"), and checkboxes for "Метод:" (with options "чтение", "запись", "вызов"). To the right of these is a "фильтры:" section with a "Убрать" button. At the very bottom, there are buttons for "Добавить", "Обработать", "Очистить всё", a text input for "Граф наследования:" (set to "classes.html"), and a checked checkbox for "свернуть всё".

Эквивалентное представление в форме диаграммы потока управления в окне web-браузера

The image displays a web browser window with two main panels. On the left is a control flow graph (CFG) for the function `long fact (long n)`. The graph is a tree structure where nodes represent code blocks and edges represent control flow. The root node is a block containing a `return` statement. This `return` node contains a ternary operator node (`$n > 0 ? n * \text{fact}(n - 1) : 1$). The ternary operator node branches into a binary operator node (>) and a block node. The binary operator node (>) branches into a block node (containing the variable long n) and a call node (fact()). The call node branches into a block node (containing the variable functionproto fact) and a binary operator node (-). The block node containing int: 1 is the base case of the recursion.`

On the right is a code editor window titled `stest.cc` showing the source code of the factorial function:

```
1 long fact(long n)
2 {
3     return n > 0 ? n * fact(n - 1) : 1;
4 }
5
```

Below the code editor is a status bar showing `line: 3 / 5 col: 22 sel: 0 BCT TAB mode: Unix (LF)`.

At the bottom right, there is a settings panel with the following options:

- recursion (рекурсия)
- trash init (инициализация мусором)
- assign in condition (= в условии)
- while цикл
- for цикл

At the bottom of the browser window, there is a toolbar with buttons: `Добавить`, `Обработать`, `Очистить всё`, `Граф наследования: classes.html`, `свернуть всё`, and a search bar with `ID:`, `PID:`, `тип: Функция`, `Доступ: Любой`, `Метод: чтение запись вызов`, `фильтры: r`, and `Убрать`.

Эквивалентное представление и навигация к исходному тексту в окне web-браузера

The image displays a development environment with two main windows and a bottom control panel.

Code Tree (Left): Shows a hierarchical view of the code structure. The root is a `while` loop. Inside the loop, there is a `if` statement. The `if` statement contains a binary operator `==` and a `return` statement. Below the `if` statement is a linear block containing an assignment operator `=`.

Code Editor (Right): Shows the source code for `tools.h`. The code is as follows:

```
61         head->next=0;
62     }
63 }
64 Tool* getTool(int id)
65 {
66     tlist* it=head;
67     while(it)
68     {
69         if(it->tool->getID()==id) return it->
70         it=it->next;
71     }
72     return 0;
73 }
74 };
```

Status Bar (Bottom): Shows the current cursor position: `строка: 67 столбец: 8` and the mode: `режим: Unix (LF)`.

Control Panel (Bottom): Contains several controls for navigating and managing code elements:

- Buttons: `Добавить`, `Обработать`, `Очистить всё`, `свернуть всё`
- Fields: `ID:`, `PID:`, `Тип:`, `Доступ:`, `Метод:`, `фильтры:`, `Убрать`
- Current selection: `tools.h - 67:2 (смотреть код)` with a `закреть` button.

Файл эквивалентного представления в окне текстового редактора Geany

```
1 <!DOCTYPE HTML>
2 <html>
3 <head>
4 <meta http-equiv='Content-Type' content='text/html; charset=utf-8' />
5 <script type='text/javascript' src='/usr/local/share/conan/graph.js'></script>
6 <link rel='stylesheet' type='text/css' href='/usr/local/share/conan/graph.css'>
7 </head>
8 <body>
9
10 <div id='wrap'><div class='lineNumbers'><div class='canvas'>
11 <div class='cur_file'>File: <a href='navtext:geany:/home/mozy/tests/test/stest.cc#1' title='Open in geany'>st
12 <span class='block_title'><span class='begin_pic'></span>long fact ( long n )</span></div>
13 <div style='display:block' id='in_4245298'><div class='inner_block'><table class='block_tbl'><tr><th><span cl
14 <div style='display:block' id='in_4245307'><div class='inner_block'><table class='block_tbl'><tr><th><span cl
15 <div style='display:none' id='in_4245311'><div class='inner_block'><table class='block_tbl'><tr><th><span cla
16 <span class='block_title'><span class='cond_pic'></span>тернарный оператор</span></div>
17 <div style='display:block' id='in_4245313'><div class='inner_block'><table class='block_tbl'><tr><th><span cl
18 <span class='block_title'><span class='block_pic'></span>бинарный оператор: &gt;</span></div><div style='disp
19 <span class='block_title_n'>переменная: <b>long</b> n</span></div></div>
20 </td></tr></table><div class='block_literal' id='4245321'><div class='linenumber'><a href='navtext:geany:/hor
21 </div><br>
22 </div></div>
23 </div>
24 </td></tr></table><div>
25 <span class='close_but' onclick='close_open_block(this,"in_4245325way1");return false;'></span> <span class='
26 <div style='display:block' id='in_4245325way1'><div class='inner_block'><table class='block_tbl'><tr><th>spa
27 <span class='block_title'><span class='block_pic'></span>бинарный оператор: *</span></div><div style='display
28 <span class='block_title_n'>переменная: <b>long</b> n</span></div></div>
29 </td></tr></table><table class='block_tbl'><tr><th><span class='open_but' onclick='close_open_block(this,"in
30 <span class='block_title_n'><span class='call_pic'></span>вызов: (<b>long</b>) <b><a href='#fact'>fact</a></b>
31 <span class='block_title_n'>переменная: <b>functionproto</b> fact</span></div></div>
32 </td></tr></table><table class='block_tbl'><tr><th><span class='open_but' onclick='close_open_block(this,"in
33 <span class='block_title'><span class='block_pic'></span>бинарный оператор: -</span></div><div style='display
```


Видео демонстрация
