

Интеграция компилятора clang со сторонней библиотекой оптимизирующих преобразований

Дубров Д. В., Патерикин А. Е.

Институт математики, механики и компьютерных наук имени
И.И. Воровича

5 Апреля 2017

Оптимизирующая распараллеливающая система

Диалоговый высокоуровневый оптимизирующий распараллеливатель

File Edit View Window Help

OpenMP2.c

```
int main(int argc, char **argv)
{
    int i, j;
    float a[100][100];
    float sum=0.0;
    for (i = 1; i < 100; i = i + 1) {
        a[i][0] = a[i][0];
    }

    for (j = 0; j < 100; j = j + 1) {
        for (i=0; i<100; i=i+1)
            a[i][j] = a[i][j]/123.456;
    }
    return 1;
}
```

Transformations

- Backends
 - Filters generator
 - GPGPU code generator
 - OpenMP Generator
 - Parallel Loop Marker
 - Reprise XML
 - VHDL Generator
- Canonization
 - Alias Canonical Form
- Data Distribution
 - Block Affine Data Distribution
 - Data Distribution For Shared M
- General
 - Arithmetic Operator Expansion
 - Constant Propagation
 - Expression Simplifier
 - Remove Unused Declarations
 - Substitution Forward
 - Swap Statements
- If

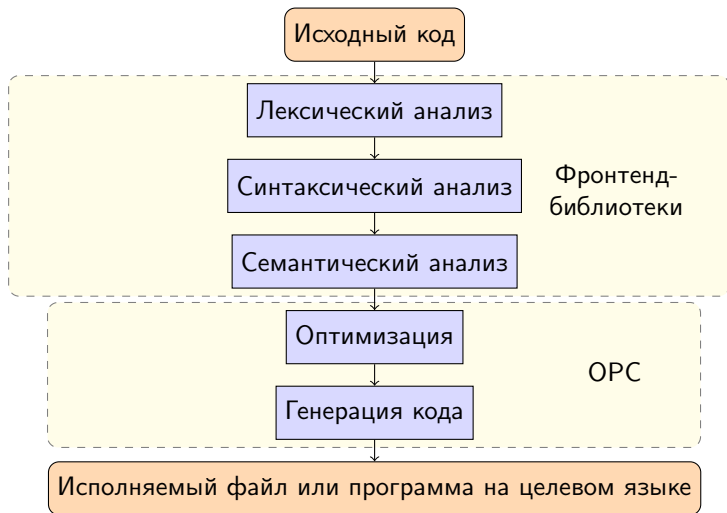
Transformation Result:

```
int main(int argc, char **argv)
{
    int i;
    int j;
    float a[100][100];
    float sum;
    sum = 0.;
    #pragma omp parallel for
    for (i = 1; i < 100; i = i + 1)
    {
        a[i][0] = a[i][0];
    }
    #pragma omp parallel for private(i)
    for (j = 0; j < 100; j = j + 1)
    {
        #pragma omp parallel for
        for (i = 0; i < 100; i = i + 1)
        {
            a[i][j] = a[i][j] / 123.456;
        }
    }
    return 1;
}
```

Generate Apply Remove unused decls Save As... As New

Editor Selector Predicates Integrity Test Transformations Graphs

Оптимизирующая распараллеливающая система



Структура OPC

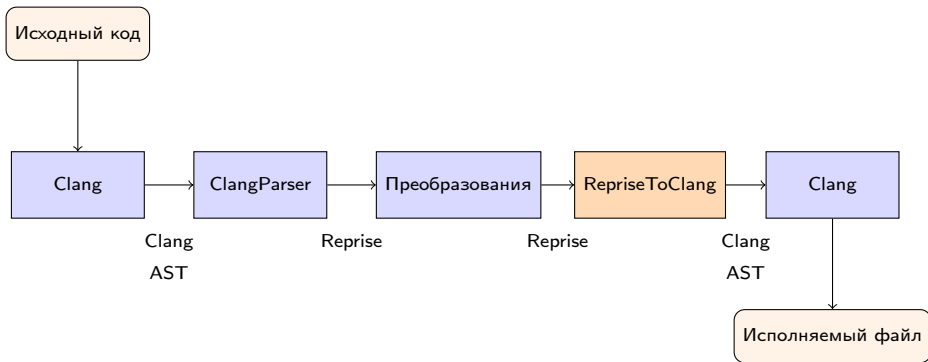
- Фронтенды ← LLVM + Clang
- Анализы
- Трансформации
- Бекенды
- Вспомогательные модули

Абстрактное синтаксическое дерево

Абстрактное синтаксическое дерево — конечное, помеченное, ориентированное дерево, в котором внутренние вершины сопоставлены с операторами языка программирования, а листья с соответствующими операндами.

Система	Внутреннее представление
clang	Clang AST
OPC	Reprise

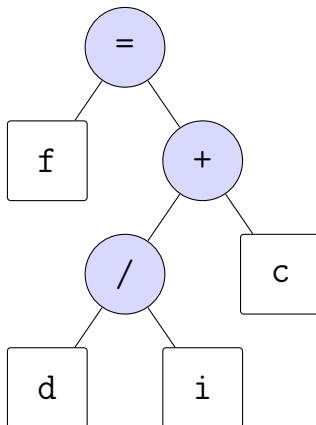
RepriseToClang



Сравнение деревьев Reprise и Clang AST

`f = d/i + c;`

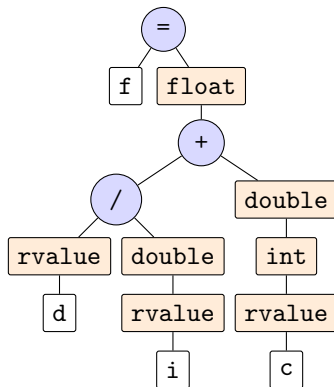
Reprise



Сравнение деревьев Reprise и Clang AST

```
f = d/i + c;
```

Clang AST



Неявные преобразования

Преобразования
lvalue к rvalue
C99 6.3.2.1

```
double d;  
char c;  
d = c;
```

Продвижение
целочисленных типов
C99 6.3.1.1 п2

```
double d;  
char c;  
d = d + (double) (int) c;
```

Обычные арифметические
преобразования
C99 6.3.1.8

```
f = (float) (d / (double) i + (double) (int) c)
```

Пример

```
int main()
{
    int i;
    float f;
    double d;
    char c;

    f = d/i + c;

    return 0;
}
```

```
f = (float)( d/(double)i + (double)(int)c )
```

Результат RepriseToClang

```
| -BinaryOperator 'float' '='  
| | -DeclRefExpr 'float' lvalue Var 'f' 'float'  
| | '-ImplicitCastExpr 'float' <FloatingCast>  
| |   '-BinaryOperator 'double' '+'  
| |     | -BinaryOperator 'double' '/'  
| |       | | -ImplicitCastExpr 'double' <LValueToRValue>  
| |         | | '-DeclRefExpr 'double' lvalue Var 'd' 'double'  
| |           | '-ImplicitCastExpr 'double' <IntegralToFloating>  
| |             | '-ImplicitCastExpr 'int' <LValueToRValue>  
| |               '-DeclRefExpr 'int' lvalue Var 'i' 'int'  
| |             '-ImplicitCastExpr 'double' <IntegralToFloating>  
| |           '-ImplicitCastExpr 'int' <IntegralCast>  
| |             '-ImplicitCastExpr 'char' <LValueToRValue>  
| |               '-DeclRefExpr 'char' lvalue Var 'c' 'char'
```

Представление операторов switch

Reprise

- Условное выражение.
- Метки case и default.

Clang

- Условное выражение.
- Метки case и default.
- Тело оператора.

Поддерживается GNU-диалект языка Си.

Представление операторов switch

Reprise

- Условное выражение.
- Метки case и default.

Clang

- Условное выражение.
- Метки case и default.
- Тело оператора.

Поддерживается GNU-диалект языка Си.

```
switch(i)
{
    case 1:  i = 2;
    default: i = 3;
}
```

```
| -SwitchStmt
| | -<<<NULL>>>
| | -ImplicitCastExpr 'int' <LValueToRValue>
| | ' -DeclRefExpr 'int' lvalue Var 'i' 'int'
| ' -CompoundStmt
| | -CaseStmt
| | | -IntegerLiteral 'int' 1
| | | -<<<NULL>>>
| | |
| | ' -DefaultStmt
| | |
| | |
```

Ассемблерные вставки

Reprise

- Формат GCC (с ограничениями).
- Формат Microsoft (с ограничениями).
- Код хранится в виде строки.

```
__asm (" mov ax,ex ");
```

Clang

- Много свойств (волатильность, список токенов, constraints, clobber list...)
- Код хранится в виде строки.

```
| -GCCAsmStmt
```

Явные преобразования типов

```
float f;          |-CStyleCastExpr 'int' <FloatingToIntegral>
char c;          | '-ImplicitCastExpr 'float' <LValueToRValue>
char* p_c;      |   '-DeclRefExpr 'float' lvalue Var 'f' 'float'
                |-CStyleCastExpr 'double' <IntegralToFloating>
                | '-ImplicitCastExpr 'char' <LValueToRValue>
(int) f;        |   '-DeclRefExpr 'char' lvalue Var 'c' 'char'
(double) c;    | -CStyleCastExpr 'int *' <BitCast>
(int*) p_c;    | '-ImplicitCastExpr 'char *' <LValueToRValue>
                |   '-DeclRefExpr 'char *' lvalue Var 'p_c' 'char *'
```

ClangInjection

