Московский физико-технический институт
(государственный университет)
Физтех-школа радиотехники и компьютерных технологий
Кафедра информатики и вычислительной техники

Выпускная квалификационная работа
(бакалаврская работа)

Оптимизация конструкции switch методом записи табличного значения

Выполнил: Балышев А. Ю., 713 гр.
Научный руководитель: к.ф.-м.н., Нейман-заде М. И.

Конструкция Switch

Термины

Цель - начало пользовательского кода под оператором case или default. (на рис. справа обозначено как «код_1», «код_2» и т.д.)

Селектор - аргумент оператора switch. (на рис. справа обозначено как «выражение»)

Значение цели - константа, расположенная в case блоке. (на рис. справа обозначено как «значение_1», «значение_2», и т.д.)

Контекст цели - множество переменных и процедур описанных в case блоке. (в цели)

```
switch (выражение)
{
  case значение_1:
    код_1;
    break;
  case значение_2:
    код_2;
    break;
  case значение_n:
    код_n;
    break;
  default:
    код;
}
```

Рис. - Представление конструкции switch в коде

Метод записи табличного значения

Причины появления метода

В одном из модулей приложения llvm была встречена функция, содержащая определенный вид конструкции switch, который характеризуется тем, что в каждой цели происходит присваивание переменной некоторого константного значения.

При компиляции данная конструкция switch раскрывалась в последовательность if-else сравнений, что приводило к долгому времени работы компилятора lcc из-за большого числа целей.

```
switch(Operand.getReg()) {
  case AMDGPU::TO_X: OpKind = MCK_R600_TReg32_X; break;
  case AMDGPU::TO_Y: OpKind = MCK_R600_TReg32_Y; break;
  case AMDGPU::TO_Z: OpKind = MCK_R600_TReg32_Z; break;
  . . . (>2000 целей)
  case AMDGPU::VGPR253_VGPR254_VGPR255: OpKind = MCK_VReg_96; break;
}
```

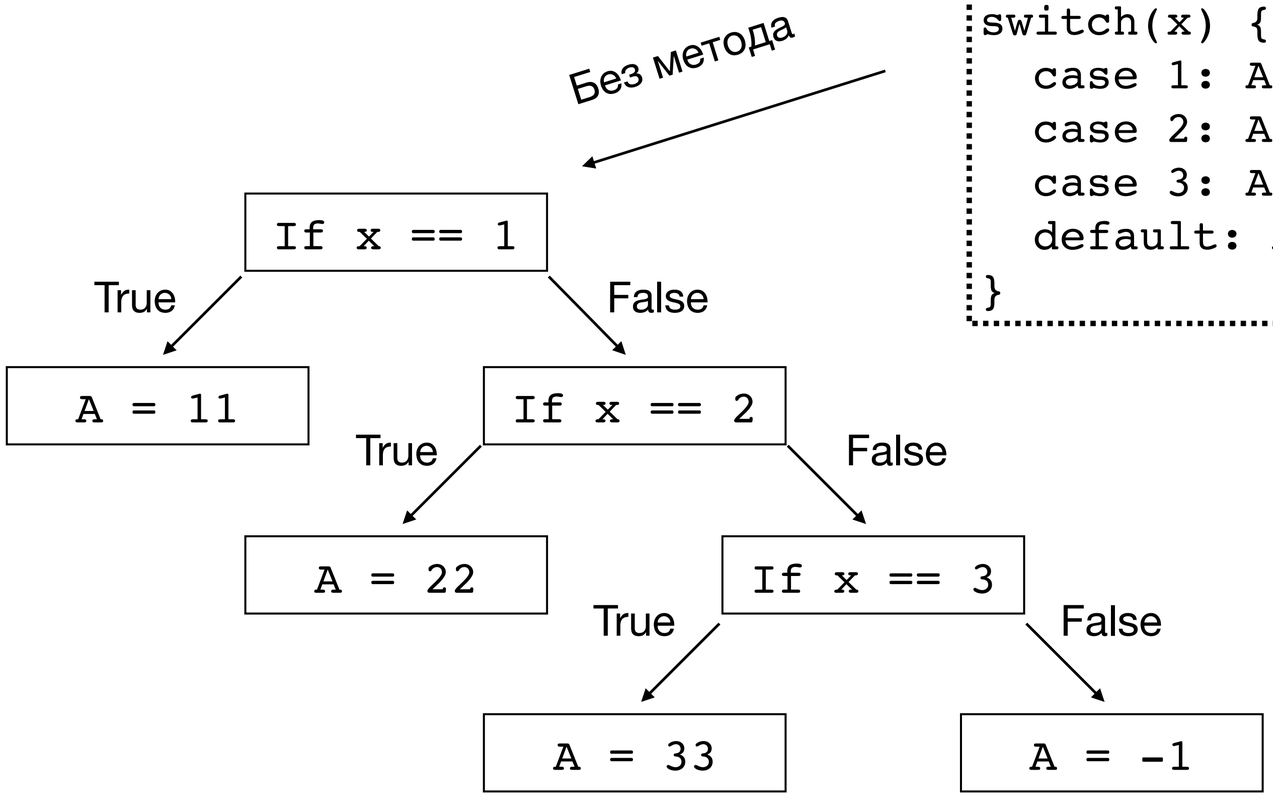
Для таких примеров конструкций switch был разработан новый метод оптимизации - **метод записи табличного значения.**

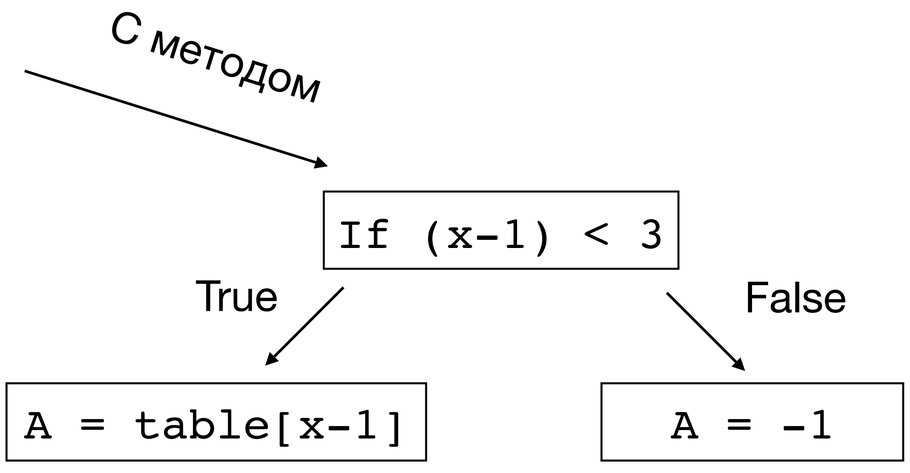
Метод записи табличного значения

Основной принцип работы метода

Идея метода заключается в том, чтобы раскрыть упомянутый пример конструкции switch через:
- таблицу, хранящую константные значения, которые присваиваются переменной в целях;
- операцию чтения (по значению селектора) из созданной таблицы;
- операцию записи прочитанного значения в переменную.

```
switch(x) {
  case 1: A = 11; break;
  case 2: A = 22; break;
  case 3: A = 33; break;
  default: A = -1;
}
```





table

индекс	данные
0	11
1	22
2	33

Цель работы

Реализация метода записи табличного значения оптимизации конструкции switch для компилятора lcc.

Задачи

- Произвести отбор примеров контекста конструкций switch, для которых можно использовать метод записи табличного значения:

  а) Исследование и анализ кодовой базы.

  б) Выбор подходящих примеров.

- В компиляторе lcc реализовать функциональность, осуществляющую оптимизацию конструкции switch с помощью метода записи табличного значения.

- Произвести измерение влияния внесенных изменений.

Исследование и анализ кодовой базы

В результате анализа кодовой базы компилятора lcc были выделены следующие варианты примеров, которые могут быть преобразованы методом записи табличного значения:

№1 Запись в переменные:

```
switch (SWITCH_VARIABLE)
{
  case  CASE_CONST_1:
    A = some_const_1A;
    B = some_const_1B;
    …
    Z = some_const_1Z;
    break;
  case  CASE_CONST_2:
    A = some_const_2A;
    B = some_const_2B;
    …
    Z = some_const_2Z;
    break;
  default:
    …
}
```

№2 Возврат констант:

```
switch (SWITCH_VARIABLE)
{
  case  CASE_CONST_1:
    return some_const_1;
  case  CASE_CONST_2:
    return  some_const_2;
    …
  case LAST_CONST:
    return  last_const;
  default:
  …
 }
```

№3 Запись с if условием:

```
switch (SWITCH_VARIABLE)
{
  case  CASE_CONST_1:
    if (BOOL_CONST_1) {
      A = some_const_1A;
      …
      Z = some_const_1Z;
    }
    break;
  case  CASE_CONST_2:
    if (BOOL_CONST_2) {
      A = some_const_2A;
      …
      Z = some_const_2Z;
    }
    break;
}
```

№4 Запись в переменную результата вызова функции:

```
switch (SWITCH_VARIABLE)
{
  case  CASE_CONST_1:
    A = some_func_1();
    break;
  case  CASE_CONST_2:
    A = some_func_2();
    break;
    …
  case LAST_CONST
    A = some_func_last();
    break;
}
```

Выбор примеров

Рассмотрим соотношения по частоте появления найденных видов примеров среди всех встреченных конструкций switch:

Пример	Количество конструкций	Процентное соотношение
№1 Запись в переменные	371	30 %
№2 Возврат констант	70	5 %
№3 Запись с if условием	19	1,6 %
№4 Запись результата функции	89	7,4 %
Все switch	1210	100 %

В результате отбора решено использовать **пример №1** (Запись в переменные) для реализации преобразования в чтение из таблицы значений.

Реализация метода записи табличного значения

Функциональность, осуществляющая оптимизацию конструкции switch с помощью метода записи табличного значения, можно разделить на две последовательные функции:

1) ответственная за отбор целей, чей контекст подходит под пример №1 «Запись в переменные» для осуществления дальнейшего преобразования.

2) осуществляющая преобразования над отобранными целями с помощью метода записи табличного значения.

Пример конструкции

```
switch(x) {
  case 1: A = 1; B = 4;
          break;
  case 2: A = 2; B = 5;
          break;
  case 3: some_func();
          break;
}
```

Функция отбора целей

Результат отбора - Список подходящих целей:

```
[
(case 1: A=1; B=4;),
(case 2: A=2; B=5;),
]
```

Функция преобразования

Результат преобразования:

A = table_A[x - 1]
B = table_B[x - 1]

table_A

индекс	данные
0	1
1	2

table_B

индекс	данные
0	4
1	5

Реализация функции отбора метода записи табличного значения

Пример конструкции

```
switch(x) {
  case 1: A = 4; B = 1;
          break;
  case 2: A = 5; B = 2;
          break;
  case 3: A = 6; B = 3;
          break;
  case 4: some_func();
          break;
}
```

Вход функции - Список целей:

```
[
(case 1: A=4; B=1;),
(case 2: A=5; B=2;),
(case 3: A=6; B=3;),
(case 4: some_func();)
]
```

Функция отбора

Выполняется проверка целей. Необходимо, чтобы:

1) Цель состояла только из операций:
- записи константы
- объявление констант

2) Проходила проверка неизменности количества переменных в цели.

3) Значения цели были последовательны

Результат отбора - Список подходящих целей:

```
[
(case 1: A=4; B=1;),
(case 2: A=5; B=2;),
(case 3: A=6; B=3;),
]
```

Реализация функции преобразования метода записи табличного значения

Результат функции отбора - Список подходящих целей:

```
[
(case 1: A=4; B=1;),
(case 2: A=5; B=2;),
(case 3: A=6; B=3;),
]
```

Функция преобразования

1) Создаёт узел проверки попадания селектора в диапазон значений цели.

2) Создаёт для каждой переменной таблицу. Таблица хранит значения из всех целей.

3) Для каждой переменной создаёт операцию чтения из таблицы по значению селектора.

4) Создаёт операции записи результата чтения для каждой переменной.

Результат функции преобразования

```
int table_A[3] = {4,5,6};
int table_B[3] = {1,2,3};
i = x - 1;
if ((unsigned)i < 3) {
  A = table_A[i];
  B = table_B[i];
}
```

Измерения влияния внесенных изменений

Описание теста

Измерения влияния внесенных изменений происходили на сгенерированных тестах.

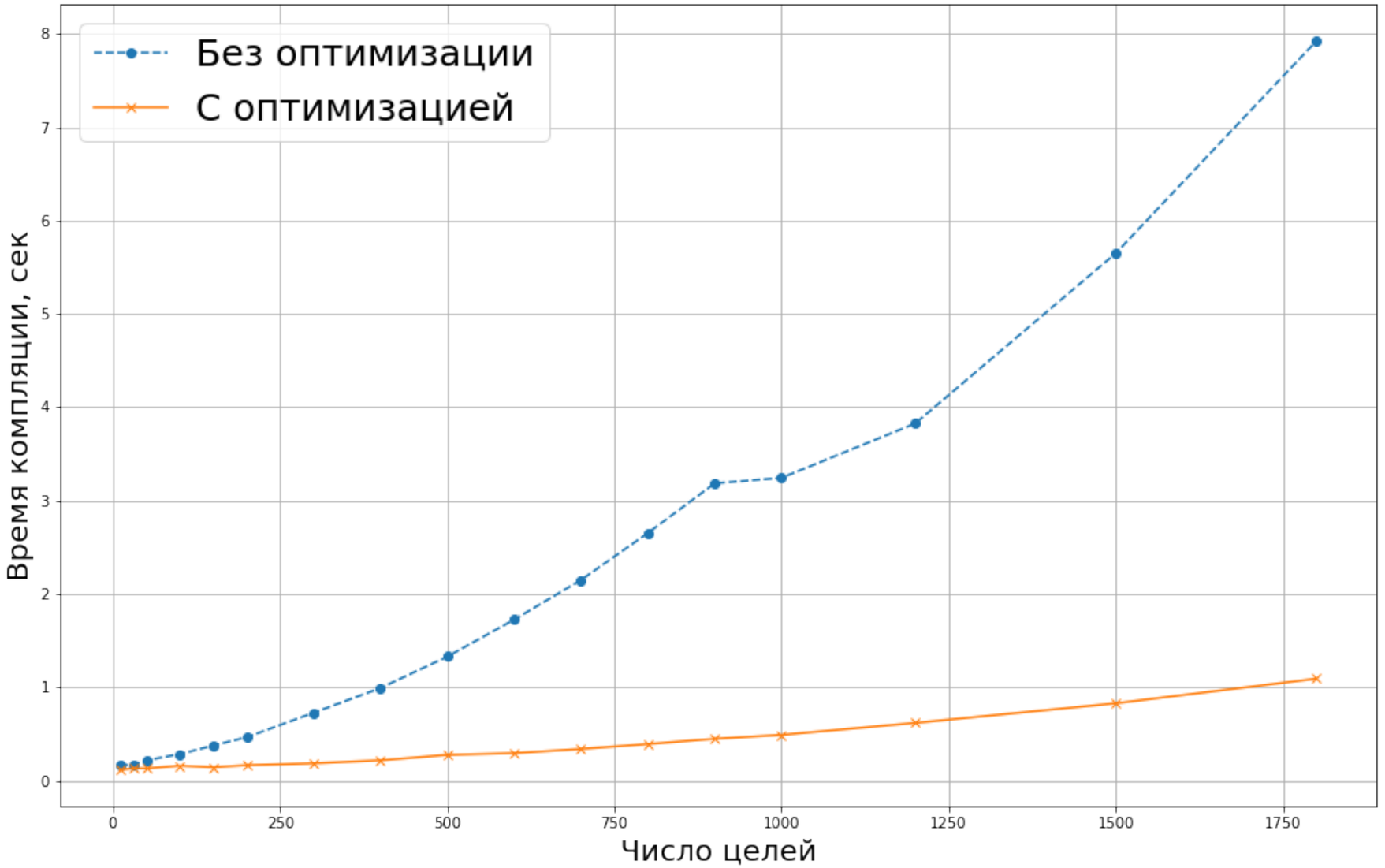
Каждый тест представляет собой программу, написанную на языке C, состоящую из функции с циклом на большое число итераций. На каждом шаге цикла выполняется конструкция switch. Эта конструкция содержит разное число целей для каждого теста. В каждом таком case блоке происходит присваивание переменной некоторого константного значения. То есть данная конструкция switch удовлетворяет определению примера №1 «Запись в переменные» и по содержанию похожа на пример конструкции из llvm, показанный в начале презентации (сл. №3).

Пример сгенерированного теста:

```
const int N = 100000000;
for(i=0; i<N; i++) {
  switch(rand()%(N)) {
    case 0: A = 0; break;
    case 1: A = 1; break;
    …
    case N-1: A = N-1; break;
    default: A = -1;
}
```

Измерения влияния внесенных изменений

Зависимость времени компиляции от числа case блоков



Результаты

- В результате проведения исследования и анализа кодовой базы был выбран пример №1 «Запись в переменные» для реализации метода записи табличного значения в компиляторе lcc.

- В компиляторе lcc реализована функциональность, осуществляющая оптимизацию конструкции switch с помощью метода записи табличного значения:
  - Реализована функция отбора целей.
  - Реализована функция преобразования отобранных целей.

- Произведены измерения внесенных изменений. Они показывают уменьшение времени компиляции на сгенерированных тестах до 8 раз.