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

ПРЕОБРАЗОВАНИЕ ПРОМЫШЛЕННЫМ КОМПИЛЯТОРОМ МАССИВА СТРУКТУР ДЛЯ ОПТИМИЗАЦИИ РАБОТЫ КЭША МИКРОПРОЦЕССОРА

Выпускная квалификационная работа
(бакалаврская работа)

Выполнил: Шампаров В. Е., 413 гр.

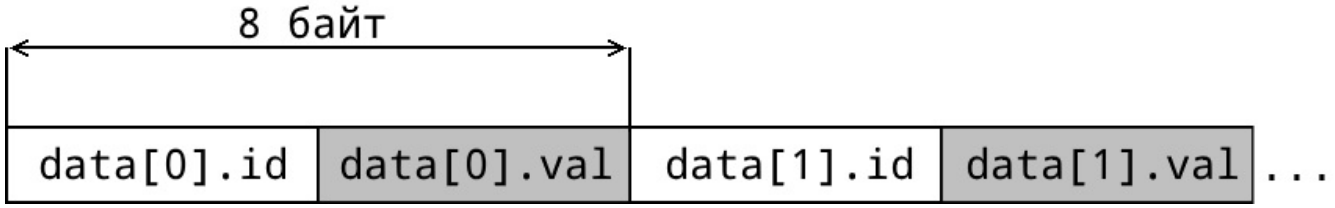
Научный руководитель: Маркин А. Л.

Проблема

```
typedef struct {int id; float val;} InnerStruct;
```

```
void UseStruct( InnerStruct *data, int size, float add) {
    int i;
    for( i = 0; i < size; i++)
        data[i].val += add;
}
```

Загружаемые в кэш данные



При работе только с одним полем в цикле часть загруженных данных не используется, что приводит к неэффективному использованию кэша

Существующее решение

В компиляторе lcc для процессоров АО «МЦСТ» реализована оптимизация sarr2arrs:

- Декомпозирует массив структур в несколько массивов, каждый из которых соответствует полю структуры

```
struct {int id; float val;} *data;
```

```
int   *id;
float *val;
```

- Неприменима для полей, являющихся массивами структур

```
typedef struct {
    int size;
    /* Вложенный массив структур ниже не обрабатывается */
    typedef struct {int id; float val;} InnerStruct *data;
} BigStruct;
```

Цель работы

Разработать оптимизацию декомпозиции SA2AIS для компилятора lcc, преобразующую массив вложенных структур в несколько массивов в охватывающей структуре

Например:

```
typedef struct {
    int size;
    typedef struct {int id; float val;} InnerStruct *data;
} BigStruct;
```

преобразуется в:

```
typedef struct {
    int    size;
    int   *sa2ais_data_id;
    float *sa2ais_data_val;
} sa2ais_BigStruct;
```

Известные решения

Оптимизации Structure Peeling для компиляторов GCC 4.3-4.8 и Open64

- Для GCC:
  - Большая алгоритмическая сложность;
- Для Open64:
  - Не позволяет выделить в отдельные массивы более одного поля;

Реализованное решение

Алгоритм:

1) этап анализа - поиск структур для оптимизации, основанный на алгоритме для Open64;
2) этап преобразования:
    1) коррекция типов;
    2) коррекция типов объектов;
    3) изменение всех обращений к скорректированным объектам;

Этап анализа

Производится поиск всех операций доступа к полям вложенных массивов структур, и найденные операции с соответствующими охватывающими структурами записываются в специальную таблицу SAFAIS.

Пример таблицы SAFAIS

`struct BigStruct1 {`
`  InnerStruct1 *data1;`
`}`
	`BigStruct1 b11; b11->data1[i].val`
	`BigStruct1 b12; b12->data1[i].id`

`struct BigStruct2 {`
`  InnerStruct2 *data2;`
`}`
	`BigStruct2 b21; b21->data2[i].id`
	`BigStruct2 b22; b22->data2[i].val`
	`BigStruct2 b23; b23->data2[i].id`

Этап преобразования
Коррекция типов

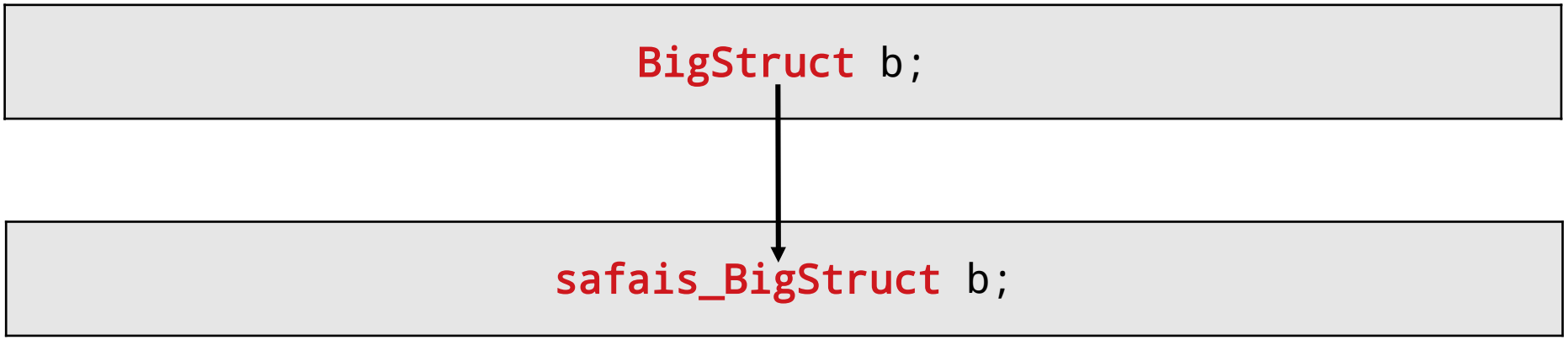
Все найденные охватывающие структуры из таблицы SAFAIS модифицируются:
вместо указателя на массив структур помещаются указатели на массивы элементов таких же типов, как типы полей вложенной структуры

```
typedef struct {
    int size;
    typedef struct {int id; float val;} InnerStruct *data;
    double other_dat;
} BigStruct;
```

```
typedef struct {
    int    size;
    int   *sa2ais_data_id;
    double other_dat;
    float *sa2ais_data_val;
} sa2ais_BigStruct;
```

Этап преобразования
Коррекция типов объектов

Заменяются типы всех объектов, имеющих тип, находящийся в таблице SAFAIS, в соответствии с модифицированными на предыдущем этапе



Этап преобразования
Замена обращений

Для всех модифицированных объектов:

1) заменяются все операции, находящиеся в таблице SAFAIS;
2) заменяется работа с массивами в следующих случаях:
    1) выделение и освобождение памяти под вложенный массив;
    2) обнуление указателя на вложенный массив;
    3) сравнение указателя на вложенный массив с нулём;

Этап преобразования
Примеры замены обращений

`b->data[i].val`	`b->sa2ais_data_val[i]`

`b->data = calloc( size,`
`  sizeof( InnerStruct));`
	`b->sa2ais_data_id  = calloc( size,`
	`  sizeof(int));`
	`b->sa2ais_data_val = calloc( size,`
	`  sizeof(float));`

`free( b->data);`	`free( b->sa2ais_data_id);` `free( b->sa2ais_data_val);`

`if( b->data == 0 ) {...}`	`if( b->sa2ais_data_id  == 0 ||`
	`  b->sa2ais_data_val == 0 )`
	`{...}`

`b->data = 0;`	`b->sa2ais_data_id  = 0;` `b->sa2ais_data_val = 0;`

Результаты
Измерения и анализ

Контекст для применения оптимизации найден в задаче 462.libquantum из пакета SPEC CPU2006.

Применение оптимизации уменьшило количество блокировок конвейера по считыванию данных в 3,46 раза, что привело к ускорению выполнения задачи на 52%.

На остальных задачах изменений производительности не зафиксировано из-за отсутствия контекста для оптимизации.

Результаты

- Реализована оптимизация SA2AIS, декомпозирующая массив вложенных структур.
- Оптимизация прошла проверку на некоторых задачах пакетов тестирования SPEC CPU. Зафиксировано ускорение выполнения на 52% на задаче 462.libquantum.