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

Автоматический анализ качества работы оптимизаций в оптимизирующем компиляторе для архитектуры “Эльбрус”

Выпускная квалификационная работа
(магистерская диссертация)

Выполнил: Левченко Д.Н., группа М01-103а
Научный руководитель: к.т.н. Ермолицкий А.В.

Введение: актуальность проблемы

Проблемы работы оптимизаций:

1. Неправильное предсказание эвристиками параметров исполнения кода, которые существенно отличаются от реального исполнения.
2. Пользователь неверно расставил подсказки для компилятора.
3. Пользователь не подал необходимые опции, без которых не работают важные оптимизации.

Введение: существующие решения

Существующий метод решения проблем:

1. Профилирование исполнения программы и поиск «горячих» участков.
2. Просмотр всех горячих участков и поиск неоптимальностей в работе оптимизаций.
3. Коррекция опций сборки задач.
4. Расстановка подсказок в коде (“прагмы” для циклов, атрибуты для функций).

Недостатки метода:

1. Необходима высокая квалификация программиста.
2. Метод имеет высокую трудоёмкость.
3. Для задач, у которых число горячих участков кода велико, метод практически не применим.

Цель работы

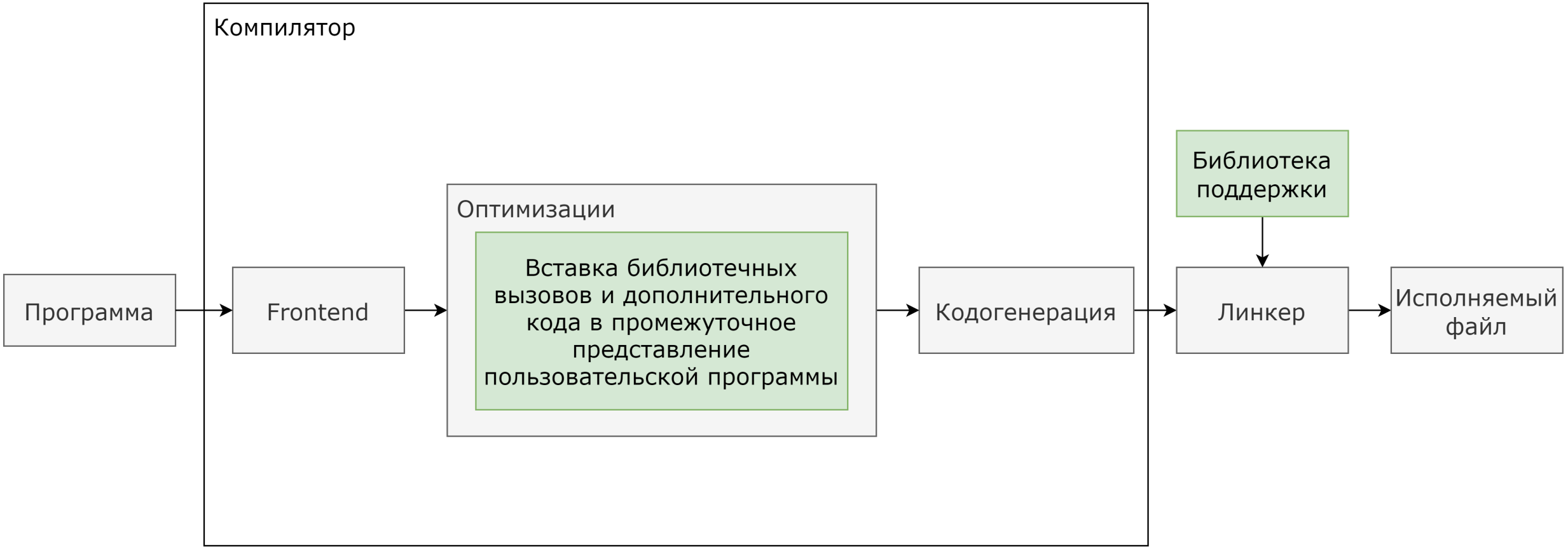
Цель работы:

Разработка в языковом компиляторе LCC инструмента для поиска неоптимальностей в работе оптимизаций inline и overlap во время работы пользовательской программы.

Задачи:

1. Разработать модуль в языковом компиляторе LCC для вставки инструментирующего кода в пользовательскую программу.
2. Разработать библиотеку поддержки для сбора статистики о проблемах оптимизаций во время работы пользовательской программы.
3. Отладить работу инструментирования на коротких направленных тестах, самосборке языкового компилятора и задачах SPEC CPU (95, 2000, 2006, 2017).
4. Выполнить самосборку компилятора LCC с разработанным инструментированием. Проанализировать полученные отчёты и расставить подсказки в целях ускорения компиляции.
5. Разработать анализатор отчётов, который, в рамках одной задачи будет выдавать суммарную статистику по этой задаче.
6. Проанализировать отчёты по задачам SPEC CPU 2017. По возможности внести корректировки в алгоритмы оптимизаций.
7. Измерить накладные расходы на инструментирование.

Общая схема инструментирования программы: компиляция

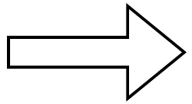


Общая схема инструментирования программы: исполнение

Принцип работы оптимизации inline

Подстановка функции в место её вызова

```
int function( void)
{
 ...
 int c = 10;
 int s = sum( c, c);
 ...
}
```

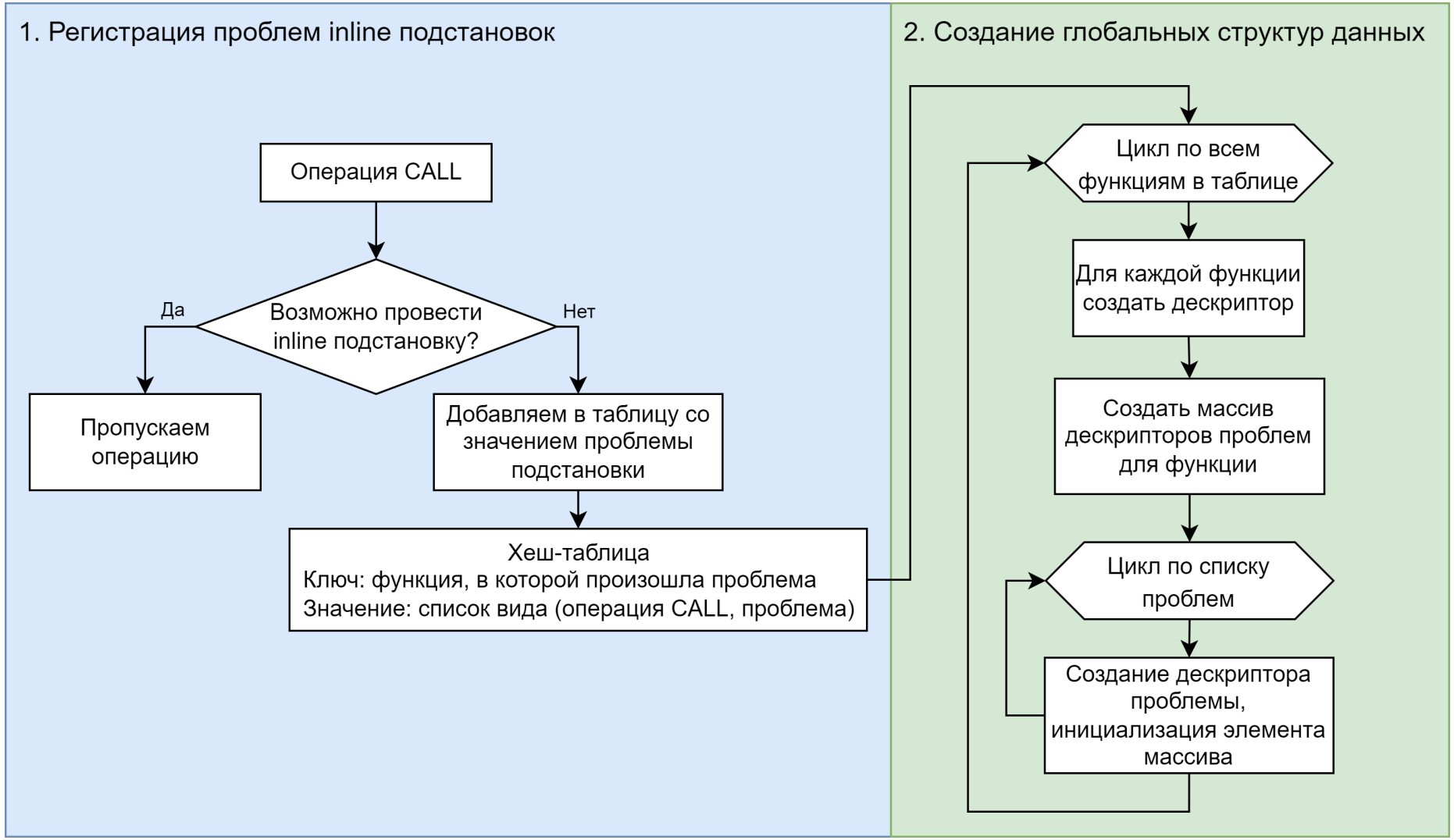


```
int function( void)
{
    ...
    int c = 10;
    int s = c + c;
    ...
}
```

```
int sum( int a, int b)
{
 return ( a + b);
}
```

Алгоритм инструментирования оптимизации inline

Первая часть инструментирования



Поля в дескрипторах

Поля дескриптора функции
Массив дескрипторов проблем внутри функции
Имя вызывающей функции
Количество операций в функции
Различные флаги

Поля дескриптора проблемы
Счётчик вызова
Имя вызываемой функции
Привязка к исходному коду (строка, модуль)
Причина, по которой не применилась оптимизация inline

Алгоритм инструментирования оптимизации inline

Вторая часть инструментирования

Пример отчёта инструментированной оптимизации inline

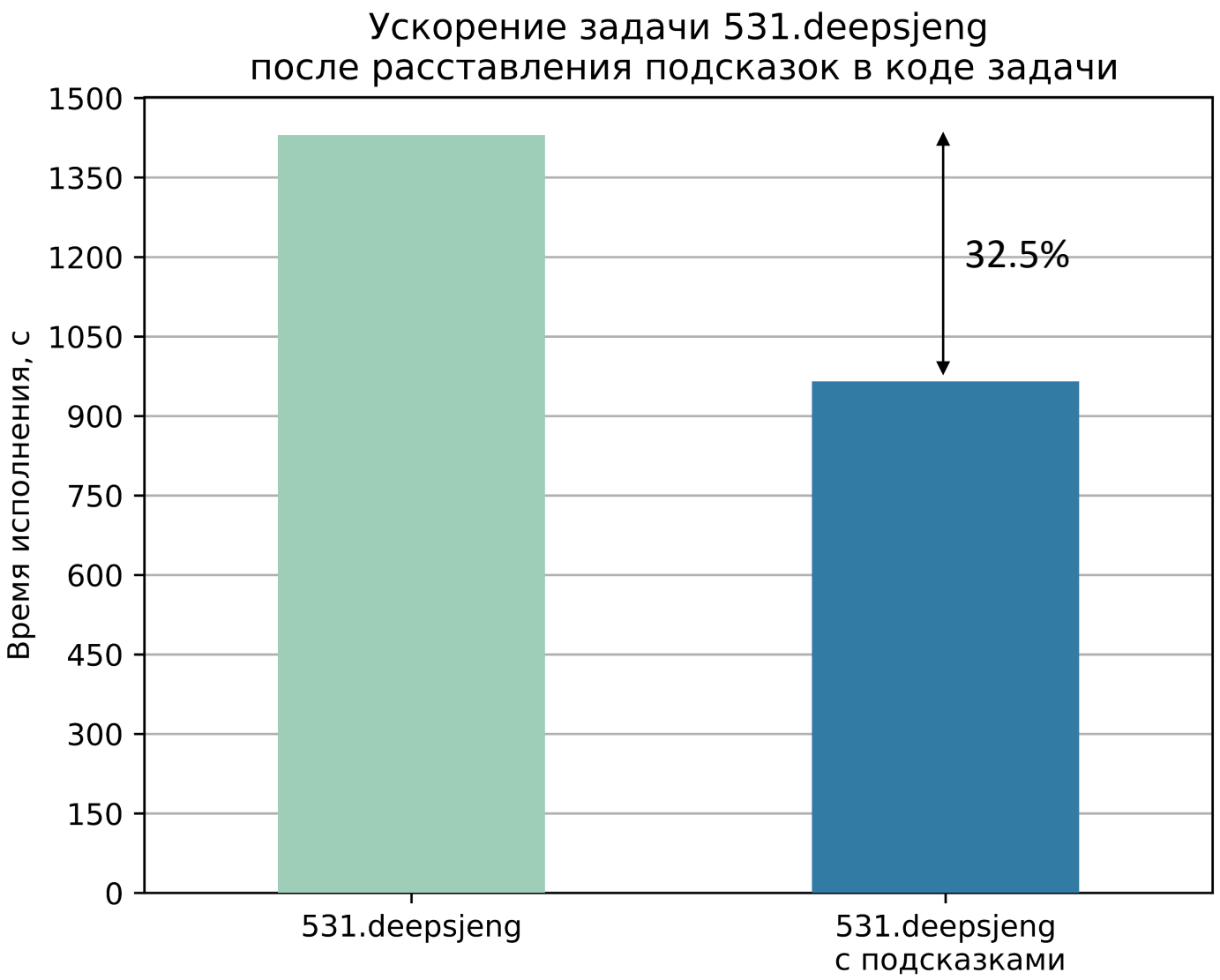
Function src name: "compact_move"
Function size: 26
Total rate: 24596094522.00 (**sum calls: 3197492288**) (**30.62 %**)
Called from:
Function src name: "order_moves"
Func size: 188
Rate: 16027850884.00 (num calls: 2083620615) (19.95 %)
Src: src/search.cpp 246
Inline fail kind: (1) Call from other module **(-fwhole required for inline)**

…
Function src name: "positional_eval"
Function size: 151
Total rate: 381469364.00 (**sum calls: 288009370**) (**0.47 %**)
Called from:
Function src name: "feval"
Func size: 173
Rate: 381469364.00 (num calls: 288009370) (0.47 %)
Src: src/neval.cpp 1081
Inline fail kind: (31) Heuristic decided inline unprofitable
[SUGGESTION] use attribute always_inline

$$Rate \sim \frac{num\_calls}{func\_size}$$

num_calls – количество вызовов функции.
func_size – количество операций в функции (её размер).

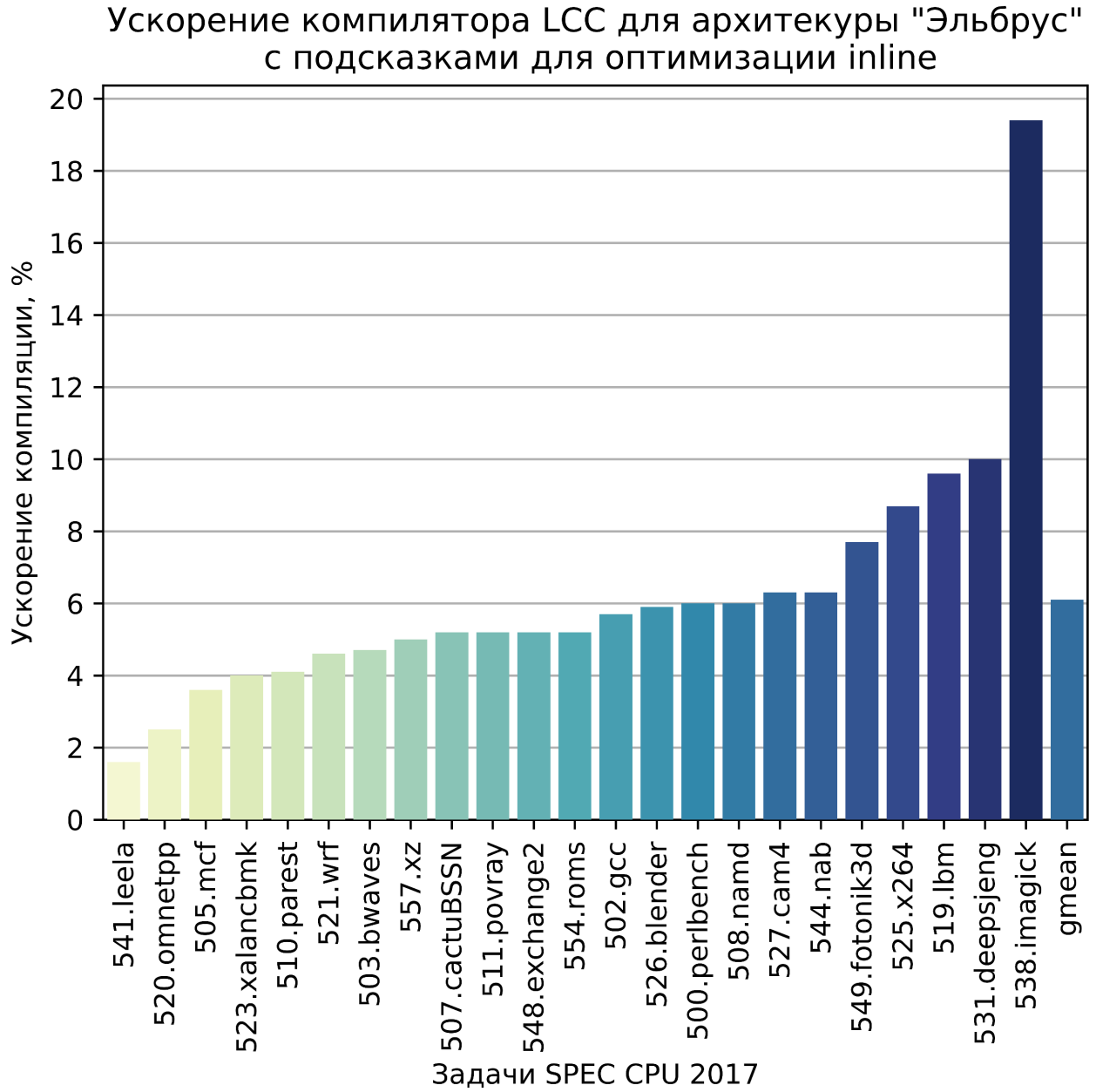
Результат расстановки подсказок в коде задачи 531.deepsjeng



В соответствии с отчётом:

- Задача была собрана с режимом –fwhole
- В коде были расставлены подсказки вида __attribute__((always_inline)) для 6 функций

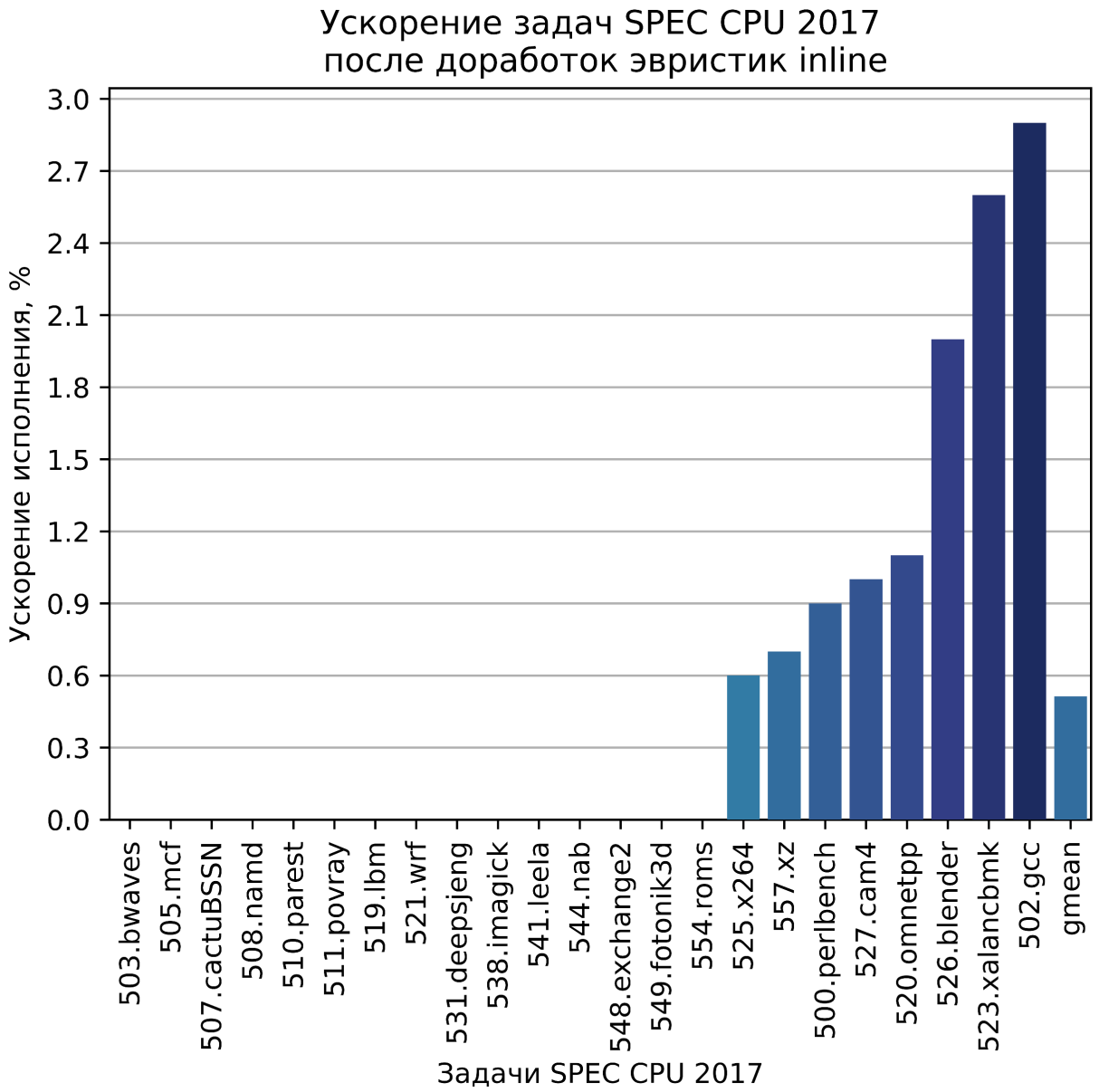
Результаты применения рекомендаций инструментирования к компилятору LCC



В соответствии с отчётом:

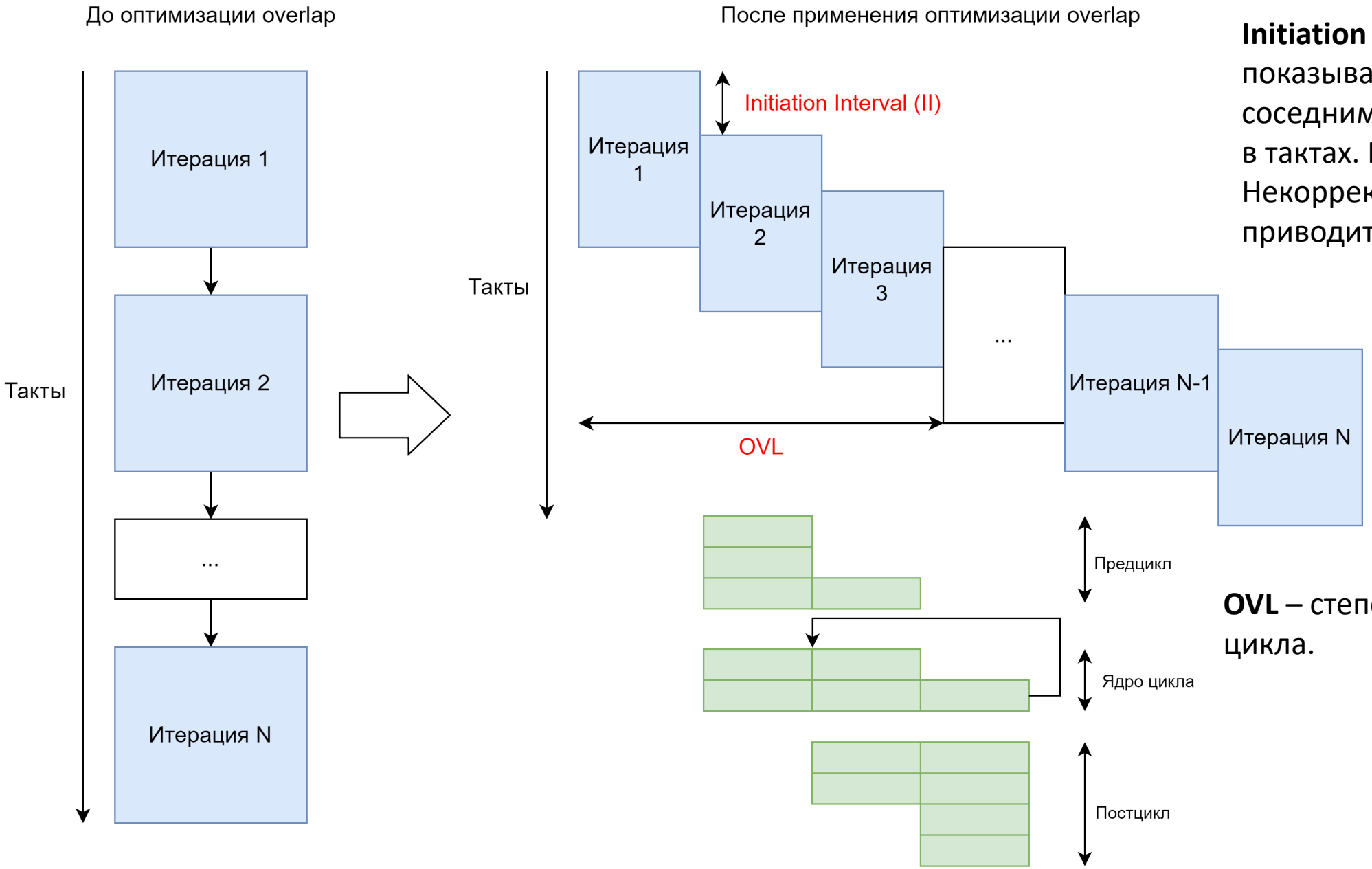
- Компилятор был собран с режимом –fwhole
- В коде были расставлены подсказки вида __attribute__((always_inline)) для 22 функций

Улучшение эвристик оптимизации inline



В результате анализа отчётов инструментирования задачи 502.gcc были обнаружены неточности в работе эвристик фазы inline. Их доработка позволила ускорить задачи SPEC CPU 2017 на 0.5% в среднем

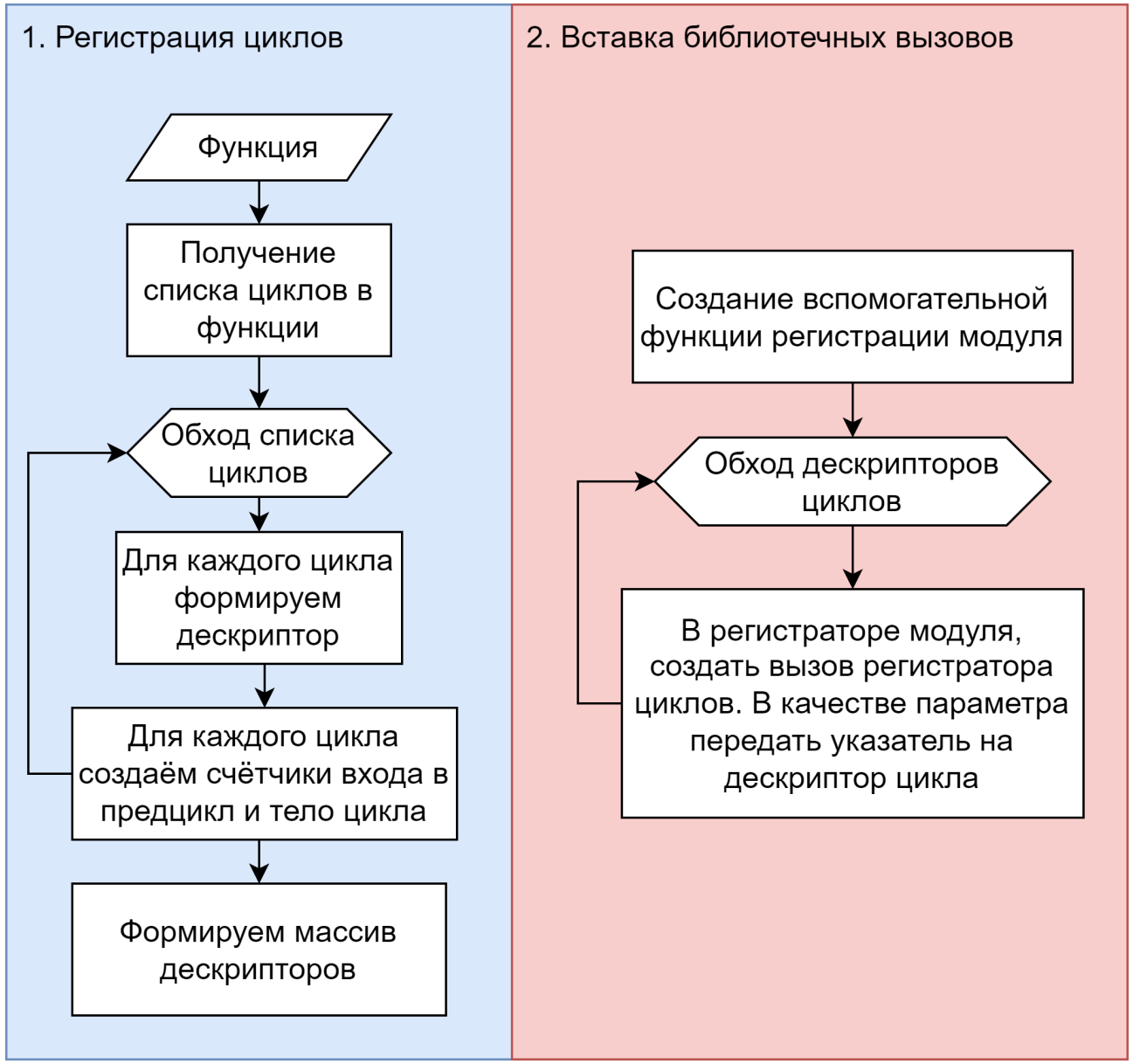
Схема работы оптимизации overlap



Initiation Interval (II) – параметр, показывающий расстояние между соседними итерациями цикла в тактах. Подбирается итеративно. Некорректный выбор может приводить к замедлению.

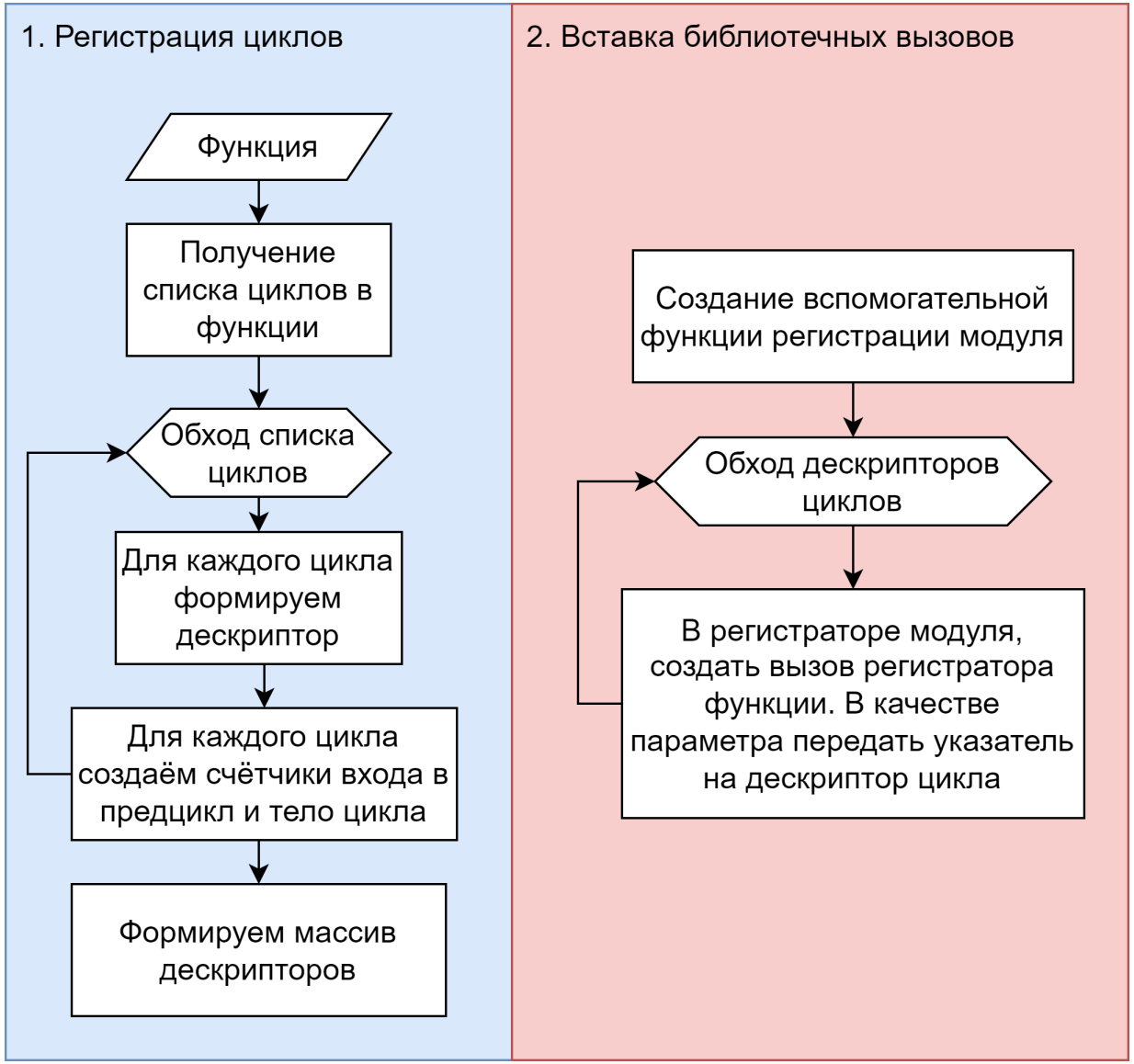
OVL – степень наложения итераций цикла.

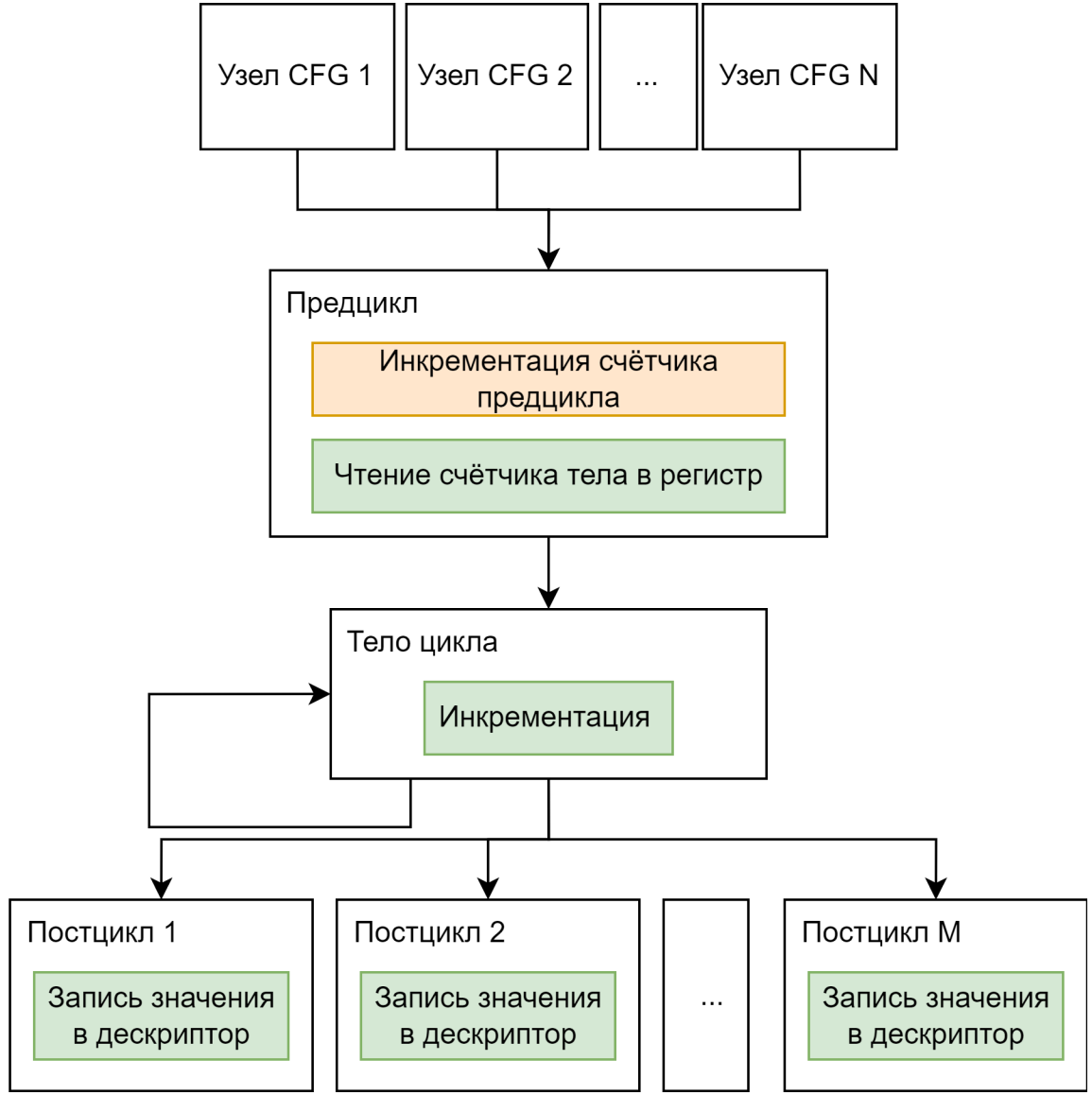
Схема инструментирования оптимизации overlap



Поля дескриптора цикла
Время исполнения: предцикла, постцикла, тела цикла (для тех, где не применилась оптимизация).
Предсказанное число итераций.
Счётчики количества входа в предцикл и в тело цикла
Привязка к исходнику (имя процедуры, строка, имя модуля, отслеживание инлайна).
Указатель на дескриптор двойника цикла
Для циклов, к которым применилась оптимизация: II, OVL.
Фактор раскрутки цикла (оптимизация unroll).
Статус применения оптимизации (была применена, не была применена, в процессе применения произошёл откат применения)

Схема инструментирования оптимизации overlap





Пример отчёта инструментированной оптимизации overlap

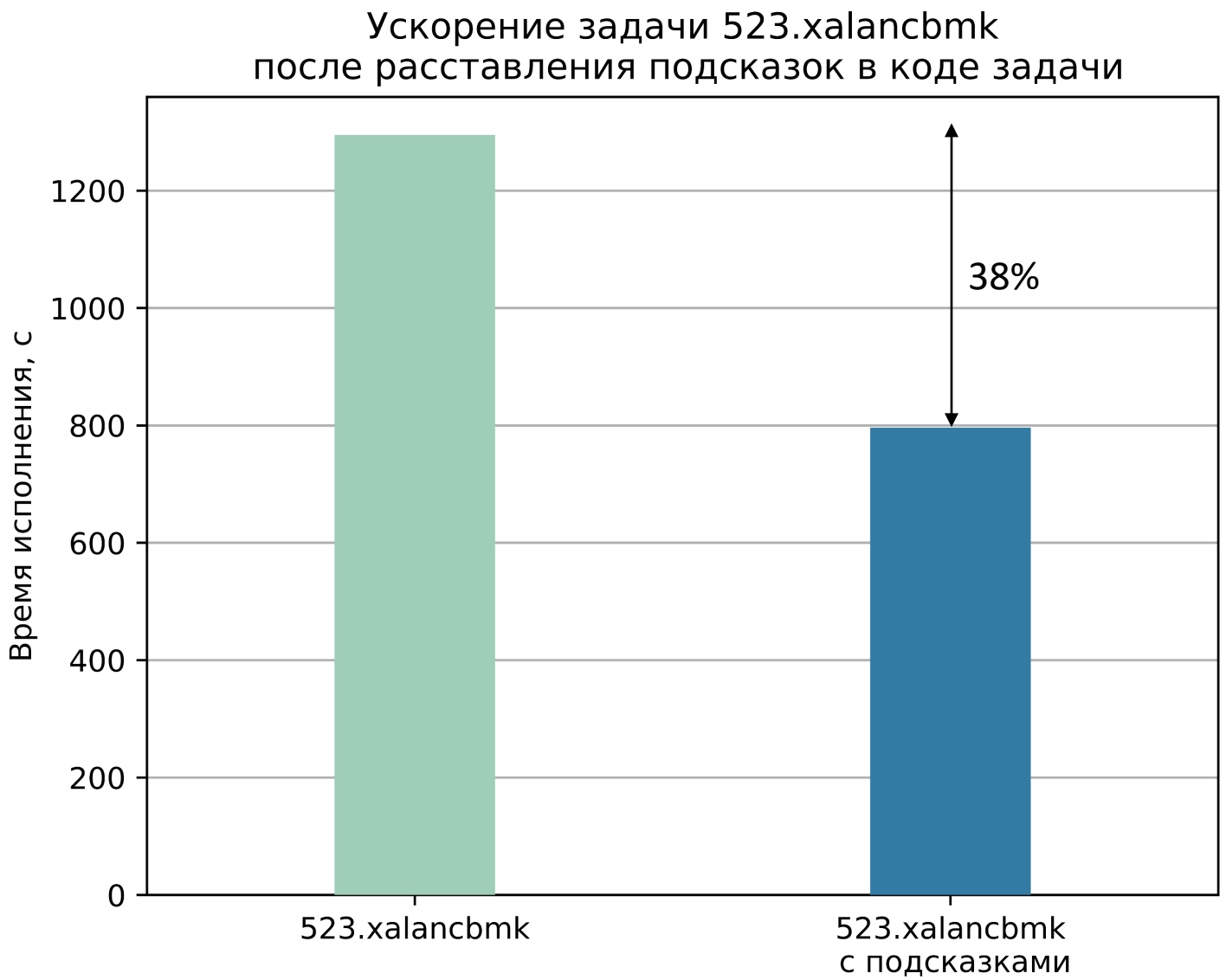
Loops for proc:
"ReusableArenaAllocator<XStringCached>::destroyObject(XStringCached*)"
proc sum weight: **217326776981.00 (37.20%** overall)

1. weight: **217247451525.00** (**99.96%** of proc, **37.18%** overall)
   src: ReusableArenaAllocator.hpp 98-128
   num loop start: 67113825
   num iterations: 16752607655
   avg. num iterations: 249.61
   Tpre = 0.00, **Tpost** = 0.00, **Tloop** = 13.00
   avg. loop exec time: 3237.00
   is loop overlapped: False
   [SUGGESTION] because of low predicted iterations num 3.00
   overlap optimization didn't apply.
   To apply overlap optimization try "#pragma loop count(N)"
   Suggested N: 250

…

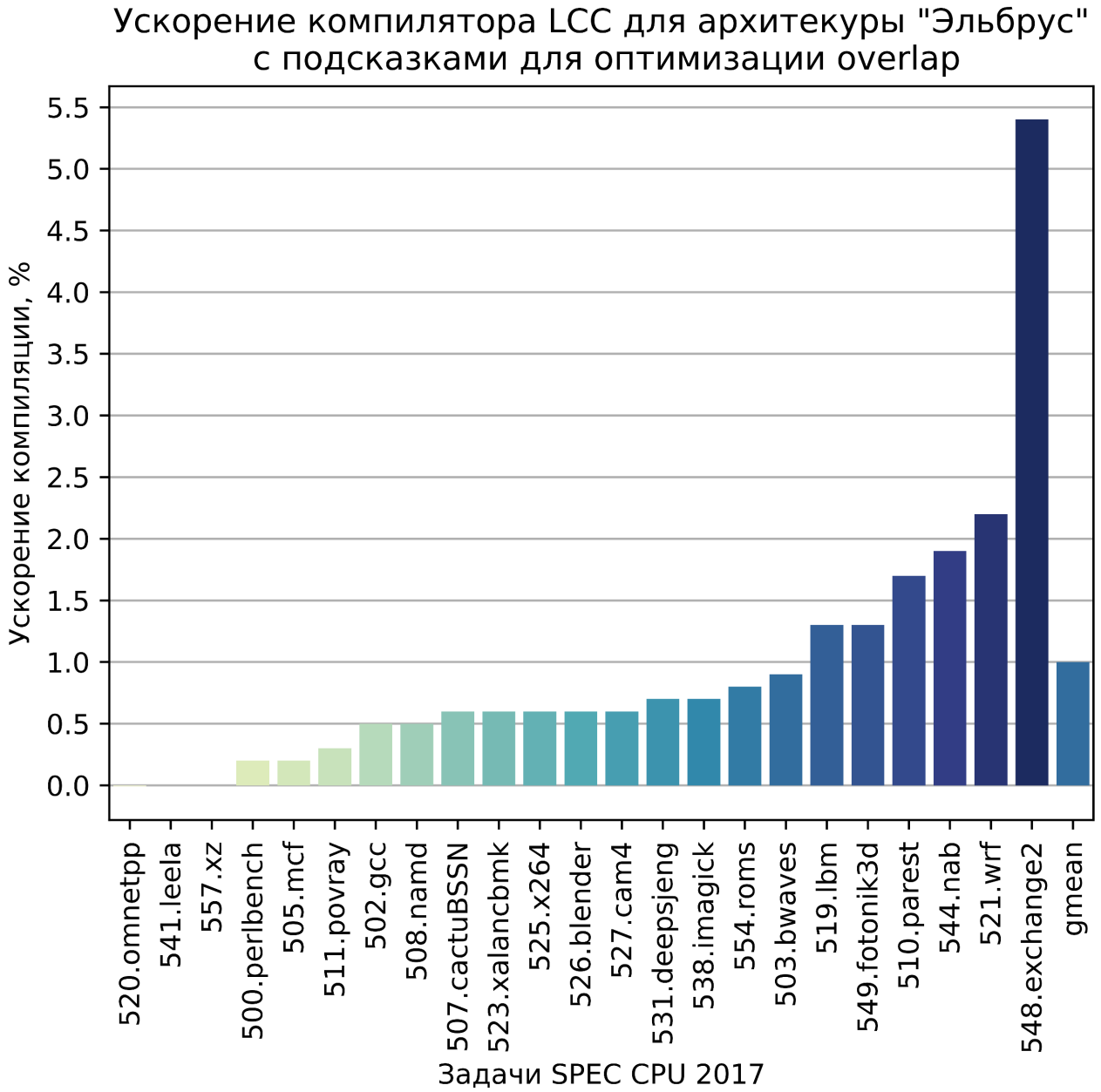
1. weight: **446472820.00** (100.00% of proc, 0.08% overall)
   src: XalanVector.hpp 323-329
   num loop start: 22323641
   num iterations: 22323641
   avg. num iterations: 1.00
   Tpre = 6.00, **Tpost** = 4.00
   II = 5 **OVL** = 2
   avg. loop exec time: 20.00
   is loop overlapped: True
   parameters for twin loop:
      Tpre = 5.00, **Tpost** = 1.00, **Tloop** = 8.00
      twin loop exec time: 14.00
   [SUGGESTION] exec time of overlapped loop
   is bigger than not overlapped more by 10%.
   Use "#pragma noswp" to disable overlap optimization.

Результат расстановки подсказок в коде задачи 523.xalancbmk



В соответствии с отчётом:
- Для 12 циклов в задаче были расставлены подсказки вида #pragma loop count(N)

Результаты применения рекомендаций инструментирования к компилятору LCC



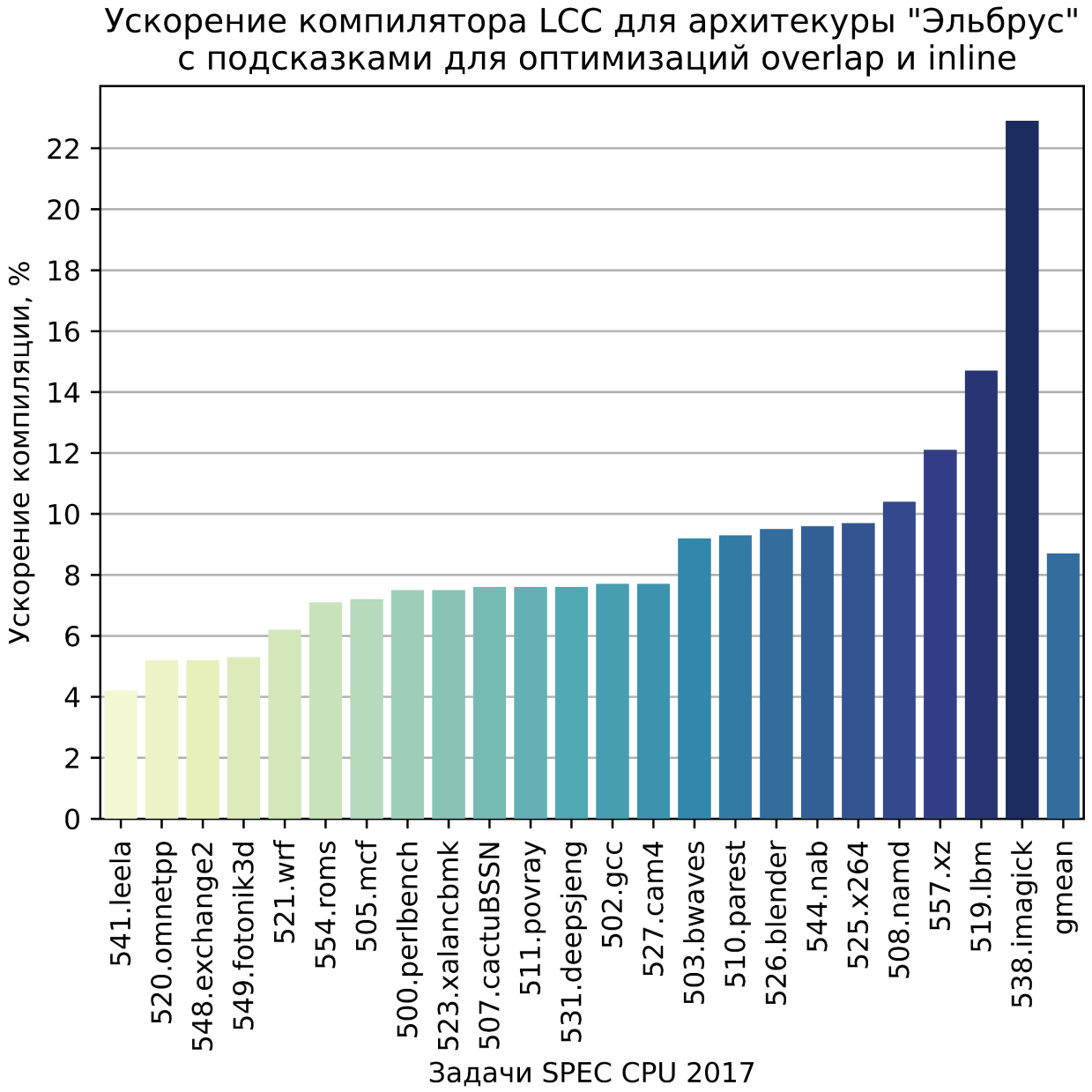
В соответствии с отчётом:

- Для 148 циклов в задаче были расставлены подсказки вида

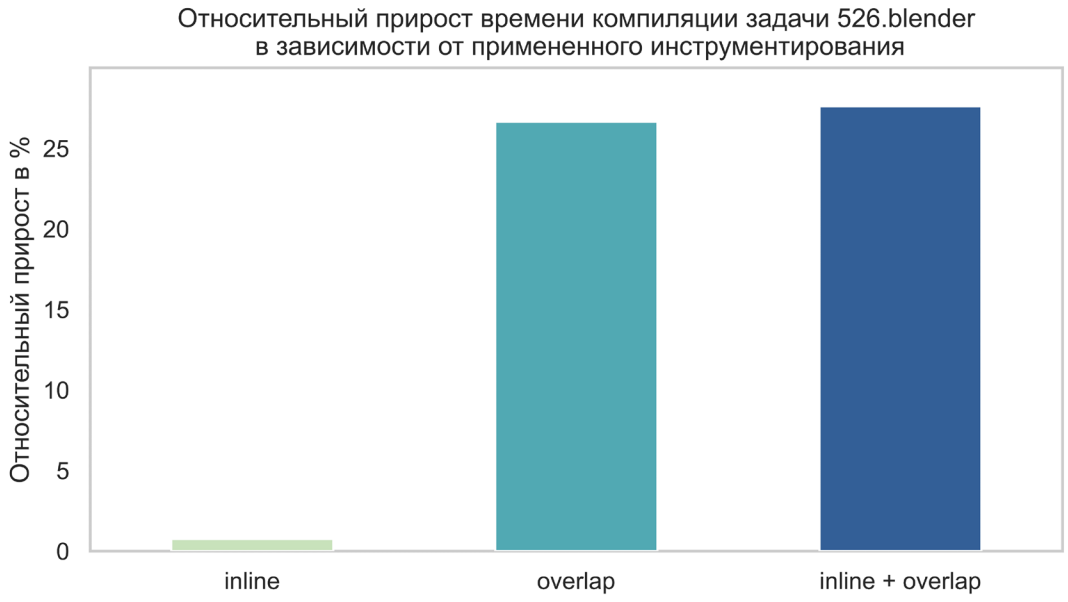
#pragma loop count(N)

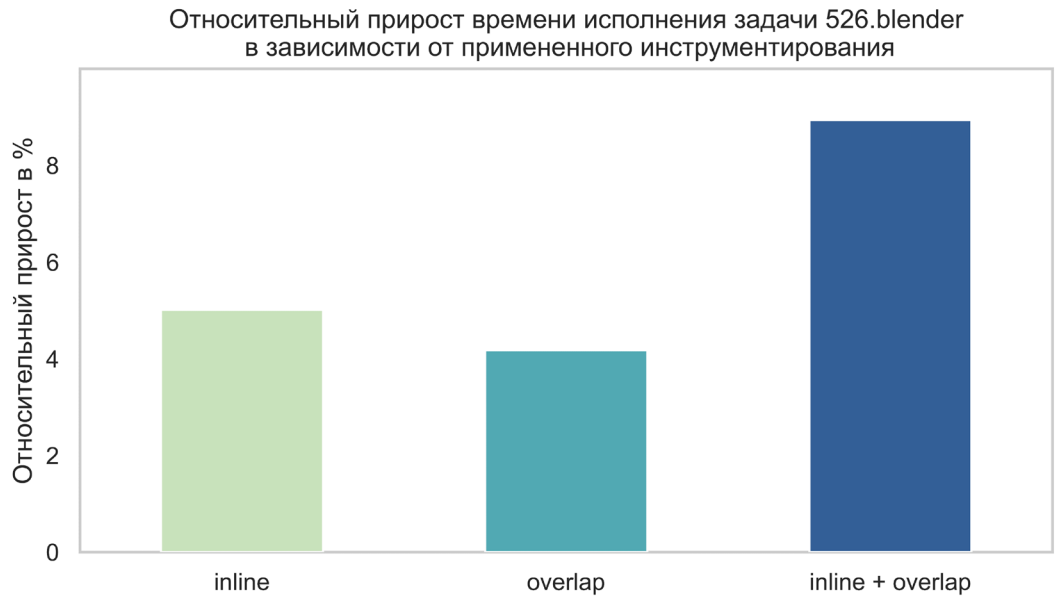
#pragma noswp

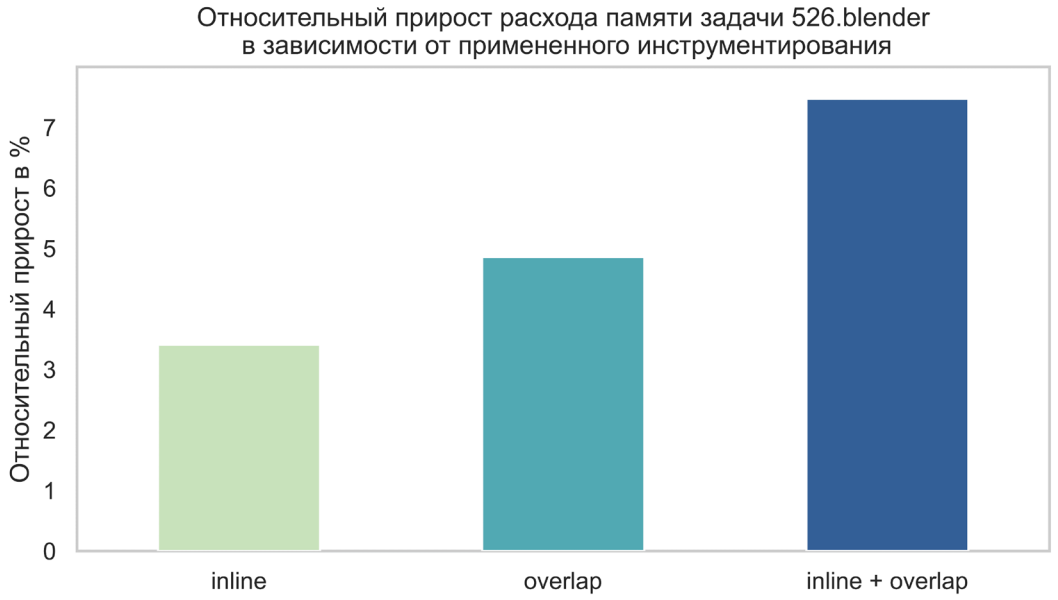
Суммарное ускорение компилятора LCC



Накладные расходы на инструментирование

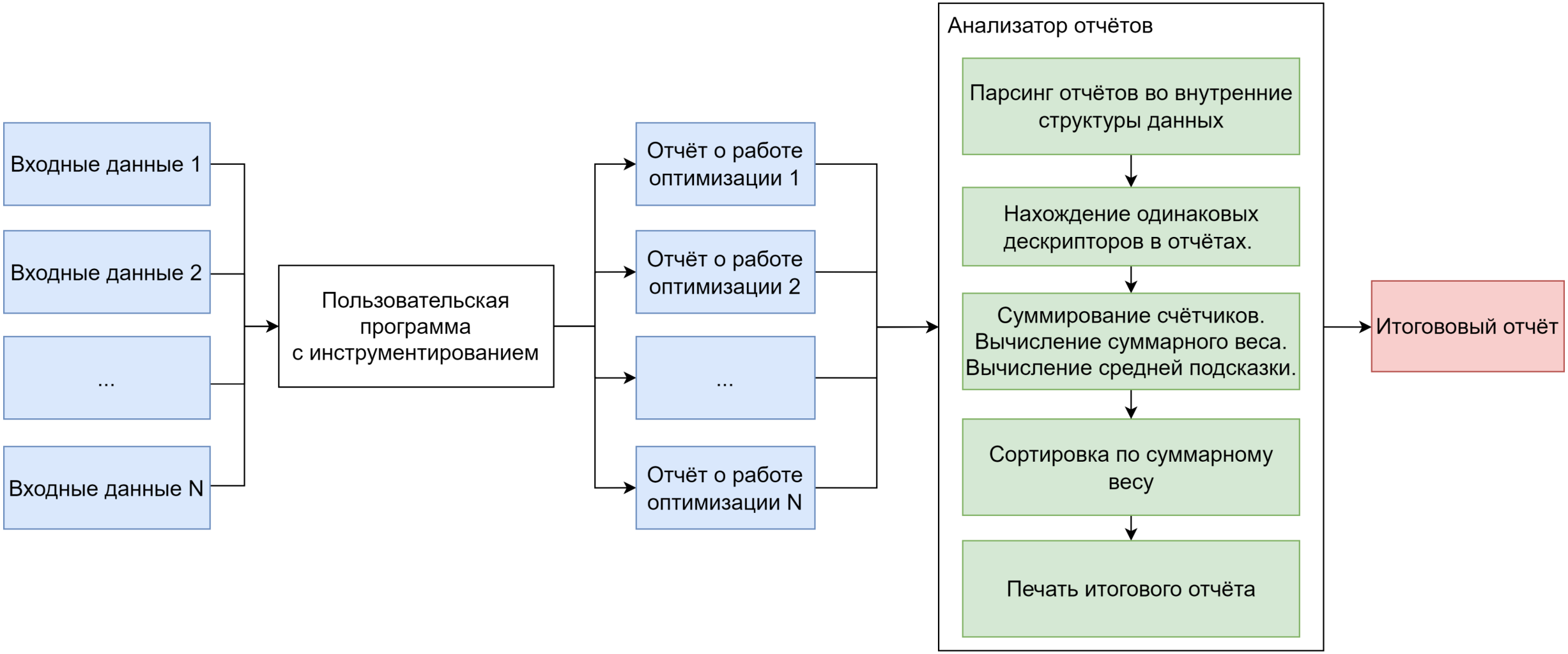






Анализатор отчётов

Схема работы



Результаты

1. Разработан модуль в языковом компиляторе LCC для вставки инструментирующего кода в пользовательскую программу для оптимизаций inline и overlap.
2. Разработана библиотека поддержки для сбора статистики о проблемах оптимизаций во время работы пользовательской программы.
3. Отлажена работа инструментирования на коротких направленных тестах, самосборке языкового компилятора и задачах SPEC CPU (95, 2000, 2006, 2017).
4. Выполнена самосборка компилятора LCC для самоинструментирования. По результатам анализа отчётов расставлены подсказки в коде, которые ускорили работу компилятора на 8.2%.
5. С помощью инструментирования ускорены задачи 531.deepsjeng и 523.xalancbmk на 32.5% и 38% соответственно.
6. По результатам инструментирования внесены правки в эвристики работы оптимизации inline. Ускорение задач SPEC CPU 2017 составило 0.5% в среднем.
7. Разработан анализатор отчётов, который в рамках одной программы для разных входных данных формирует итоговый отчёт.
8. Измерены накладные расходы на инструментирование. Прирост потребления ресурсов при исполнении инструментированной программы незначительный.

Приложение: опции инструментирования

Опции компиляции:

- -fsanitize=opt – включение инструментирования во время компиляции задачи.
- -fsanopt-report-dir= – путь до директории, куда будут попадать отчёты.
- -fsanopt-analyze={all, ovl, inline} – выбор оптимизации для анализа.

Переменные окружения (во время исполнения задачи):

- SANOPT_REPORT_NUM_PRINTS – ограничение на количество дескрипторов, выводимых на печать.
- SANOPT_REPORT_THRESHOLD_RATE – коэффициент для ограничения минимального веса, выводимого на печать.
- SANOPT_REPORT_FILE_SUFFIX – суффикс, который можно добавить к каждому отчёту

Приложение: анализатор отчётов

Вид итогового отчёта для оптимизации inline

```
Weight:  419453952132.00 (7.172%)
    Proc name: set_GetNextRef(void**)
    Proc size: 11
    Num calls: 922798695080
    ---------------------------------------------------------------------------------------------------
    |      Inline problem place      |     Source file and line | Size |       Fail kind        | Overall weight % |
    ---------------------------------------------------------------------------------------------------
    | ipa_UniteAllBaseObjsRec        | ipa/ipa_objalias.c 2731  | 153  | Call from other module |      7.168%      |
    | dfa_DeleteOptimisticConClasses | dfa/dfa_valnum.c 2400    | 186  | Call from other module |      0.003%      |
    ---------------------------------------------------------------------------------------------------

    (1) Call from other module: 99.982%
[SUGGESTION] use option -fwhole
```

Приложение: анализатор отчётов

Вид итогового отчёта для оптимизации overlap

```
Weight: 371719508737.00 (3.963%)
    Proc name: list_FindRef(void*, void*)
    Src: list.c 3958-3958
    |                    File                     | Line | Avg. iters | Suggest? | Exec weight |
    | overlap_ecf_opt64_e_521.wrf_486.report       |  4   |   47.56    |   True   |   79.30%    |
    | overlap_ecf_opt64_e_507.cactuBSSN_438.report |  15  |   28.95    |   True   |    9.57%    |
    | overlap_ecf_opt64_e_526.blender_1041.report  | 1268 |    3.80    |  False   |    3.19%    |
    | overlap_ecf_opt64_e_510.parest_233.report    | 298  |    1.68    |  False   |    2.11%    |
    | overlap_ecf_opt64_e_502.gcc_387.report       | 1553 |    5.75    |  False   |    1.26%    |
    | overlap_ecf_opt64_e_523.xalancbmk_755.report | 568  |    1.68    |  False   |    0.91%    |
    | overlap_ecf_opt64_e_554.roms_233.report      | 111  |   10.11    |   True   |    0.89%    |
    | overlap_ecf_opt64_e_508.namd_15.report       | 237  |   11.21    |   True   |    0.64%    |
    | overlap_ecf_opt64_e_525.x264_98.report       |  68  |    6.58    |   True   |    0.54%    |
    | overlap_ecf_opt64_e_538.imagick_99.report    | 843  |    4.84    |  False   |    0.49%    |
    | overlap_ecf_opt64_e_549.fotonik3d_19.report  | 173  |    5.97    |  False   |    0.26%    |
    | overlap_ecf_opt64_e_500.perlbench_69.report  | 1286 |    4.54    |  False   |    0.26%    |
    | overlap_ecf_opt64_e_520.omnetpp_177.report   | 953  |    1.82    |  False   |    0.19%    |
    | overlap_ecf_opt64_e_511.povray_103.report    | 1246 |    3.72    |  False   |    0.17%    |
    | overlap_ecf_opt64_e_548.exchange2_1.report   | 537  |    9.47    |   True   |    0.10%    |
    | overlap_ecf_opt64_e_557.xz_90.report         | 685  |    3.86    |  False   |    0.02%    |
    | overlap_ecf_opt64_e_503.bwaves_7.report      | 566  |    8.46    |   True   |    0.02%    |
    | overlap_ecf_opt64_e_541.leela_21.report      | 1033 |    2.96    |  False   |    0.02%    |
    | overlap_ecf_opt64_e_544.nab_20.report        | 1043 |    3.72    |  False   |    0.02%    |
    | overlap_ecf_opt64_e_519.lbm_2.report         | 291  |   14.38    |   True   |    0.02%    |
    | overlap_ecf_opt64_e_531.deepsjeng_20.report  | 1184 |    4.87    |  False   |    0.02%    |
    [SUGGESTION] use #pragma loop count(N). Suggested N: 48.
    Probability: 0.911
```