# Исследование оптимизации инвариантного кода циклов (LICM оптимизации)

## Исполнитель исследования

ФИО: Алибеков Мурад Рамазанович

Факультет: ИИТММ

Кафедра: МОСТ

Направление: ФИИТ

Номер группы: 381806-1

## Тестовая инфраструктура

CPU: Intel(R) Core(TM) i5-4690K CPU @ 3.50GHz

OS: Ubuntu 20.04 LTS

## Результаты тестирования

|  |  |  |  |
| --- | --- | --- | --- |
| Файл | Ключ оптимизации | | |
|  |  |  |
| baseline (-O0) | optimized (-O1) | optimized without LICM |
| licm\_5 | 1,27 с | 0,84 с | 0,90 с |
| licm\_10 | 1,16 с | 0,76 с | 0,84 с |
| licm\_20 | 1,28 с | 0,83 с | 0,94 с |
| licm\_div\_5 | 1,06 с | 0,21 с | 1,06 с |
| licm\_div\_10 | 2,09 с | 0,38 с | 2,10 с |
| licm\_div\_20 | 4,19 с | 0,84 с | 4,18 с |

## Анализ результатов

### Различия между версиями без оптимизации (-O0) и c оптимизацией (-O1)

1. Код функции main содержит вычисление набора значений переменных, объявленных перед циклом и в неоптимизированный версии, эти значения вычисляются при каждом запуске, а в оптимизированном – получены на этапе компиляции.
2. Внутри функции func инвариантные относительно цикла выражения в оптимизированной версии вычисляются за пределами цикла, а внутри подставляются готовые значения, в отличие от baseline версии, в которой подобные выражения вычисляются непосредственно внутри цикла.
3. В функции func предполагается передача параметров по значению, из-за этого в неоптимизированной версии происходит их копирование. В оптимизированной версии происходит передача по ссылке, видимо, потому что они не меняются в ходе цикла.
4. В оптимизированной версии вычисление инкремента и условия продолжения цикла встроено непосредственно в его тело, в неоптимизированной версии они вынесены в отдельные блоки.

### Особенности версии без LICM оптимизации[[1]](#footnote-1)

1. Отключение LICM-прохода компилятора подразумевает собой отключение вынесения инвариантного кода за пределы цикла. В остальном, эта версия аналогична стандартной полностью оптимизированной версии с ключом оптимизации -O1.

### Общий вывод

Как видно из результатов анализа производительности, время выполнения программы при отключении LICM-прохода близко к времени выполнения версии без оптимизации. Это происходит, потому что в тестовых примерах относительно большая часть кода инвариантна в циклах, и, как следствие, LICM оптимизация – наиболее значимая (по сравнению с другими оптимизациями) в данных примерах и с ней происходит значительное ускорение (особенно в примерах с делением).

## Приложение

### Приложение 1. Логи запусков программы анализа производительности для разных файлов

Performance counter stats for './build/licm\_5\_baseline' (10 runs):

1 266,43 msec task-clock # 0,998 CPUs utilized ( +- 0,11% )

210 context-switches # 0,166 K/sec ( +- 0,87% )

0 cpu-migrations # 0,000 K/sec

896 page-faults # 0,708 K/sec

4 899 442 269 cycles # 3,869 GHz ( +- 0,06% )

10 810 299 598 instructions # 2,21 insn per cycle ( +- 0,00% )

801 993 234 branches # 633,271 M/sec ( +- 0,00% )

60 265 branch-misses # 0,01% of all branches ( +- 0,24% )

1,26944 +- 0,00147 seconds time elapsed ( +- 0,12% )

Performance counter stats for './build/licm\_5\_optimized' (10 runs):

841,13 msec task-clock # 0,999 CPUs utilized ( +- 0,14% )

91 context-switches # 0,108 K/sec ( +- 0,91% )

0 cpu-migrations # 0,000 K/sec

896 page-faults # 0,001 M/sec

3 257 733 934 cycles # 3,873 GHz ( +- 0,03% )

6 007 484 066 instructions # 1,84 insn per cycle ( +- 0,00% )

401 329 948 branches # 477,130 M/sec ( +- 0,00% )

27 728 branch-misses # 0,01% of all branches ( +- 0,83% )

0,84172 +- 0,00121 seconds time elapsed ( +- 0,14% )

Performance counter stats for './build/licm\_5\_optimized\_without\_LICM' (10 runs):

896,92 msec task-clock # 0,998 CPUs utilized ( +- 0,10% )

98 context-switches # 0,110 K/sec ( +- 2,31% )

0 cpu-migrations # 0,000 K/sec

896 page-faults # 0,999 K/sec

3 467 411 775 cycles # 3,866 GHz ( +- 0,04% )

10 007 580 974 instructions # 2,89 insn per cycle ( +- 0,00% )

401 349 221 branches # 447,473 M/sec ( +- 0,00% )

28 312 branch-misses # 0,01% of all branches ( +- 1,51% )

0,89855 +- 0,00151 seconds time elapsed ( +- 0,17% )

Performance counter stats for './build/licm\_10\_baseline' (10 runs):

1 160,77 msec task-clock # 0,999 CPUs utilized ( +- 0,11% )

125 context-switches # 0,108 K/sec ( +- 2,40% )

0 cpu-migrations # 0,000 K/sec

1 186 page-faults # 0,001 M/sec

4 476 581 848 cycles # 3,857 GHz ( +- 0,03% )

9 409 160 993 instructions # 2,10 insn per cycle ( +- 0,00% )

401 621 914 branches # 345,996 M/sec ( +- 0,00% )

32 600 branch-misses # 0,01% of all branches ( +- 1,31% )

1,16202 +- 0,00126 seconds time elapsed ( +- 0,11% )

Performance counter stats for './build/licm\_10\_optimized' (10 runs):

756,42 msec task-clock # 0,999 CPUs utilized ( +- 0,11% )

80 context-switches # 0,106 K/sec ( +- 2,78% )

0 cpu-migrations # 0,000 K/sec

1 186 page-faults # 0,002 M/sec

2 929 184 953 cycles # 3,872 GHz ( +- 0,03% )

5 008 389 238 instructions # 1,71 insn per cycle ( +- 0,00% )

201 468 704 branches # 266,344 M/sec ( +- 0,00% )

28 129 branch-misses # 0,01% of all branches ( +- 1,48% )

0,757374 +- 0,000999 seconds time elapsed ( +- 0,13% )

Performance counter stats for './build/licm\_10\_optimized\_without\_LICM' (10 runs):

843,59 msec task-clock # 0,999 CPUs utilized ( +- 0,11% )

80 context-switches # 0,095 K/sec ( +- 2,14% )

0 cpu-migrations # 0,000 K/sec

1 186 page-faults # 0,001 M/sec

3 273 570 782 cycles # 3,881 GHz ( +- 0,06% )

9 008 730 513 instructions # 2,75 insn per cycle ( +- 0,00% )

201 534 926 branches # 238,903 M/sec ( +- 0,01% )

29 475 branch-misses # 0,01% of all branches ( +- 2,15% )

0,84485 +- 0,00140 seconds time elapsed ( +- 0,17% )

Performance counter stats for './build/licm\_20\_baseline' (10 runs):

1 274,03 msec task-clock # 0,996 CPUs utilized ( +- 0,28% )

150 context-switches # 0,118 K/sec ( +- 9,36% )

0 cpu-migrations # 0,000 K/sec

2 061 page-faults # 0,002 M/sec

4 938 351 442 cycles # 3,876 GHz ( +- 0,25% )

8 713 681 286 instructions # 1,76 insn per cycle ( +- 0,00% )

202 418 391 branches # 158,880 M/sec ( +- 0,02% )

45 647 branch-misses # 0,02% of all branches ( +- 3,83% )

1,27975 +- 0,00555 seconds time elapsed ( +- 0,43% )

Performance counter stats for './build/licm\_20\_optimized' (10 runs):

832,51 msec task-clock # 0,998 CPUs utilized ( +- 0,52% )

84 context-switches # 0,101 K/sec ( +- 1,94% )

0 cpu-migrations # 0,000 K/sec

2 061 page-faults # 0,002 M/sec

3 210 712 205 cycles # 3,857 GHz ( +- 0,46% )

4 511 619 889 instructions # 1,41 insn per cycle ( +- 0,00% )

102 012 940 branches # 122,536 M/sec ( +- 0,00% )

31 320 branch-misses # 0,03% of all branches ( +- 1,18% )

0,83457 +- 0,00481 seconds time elapsed ( +- 0,58% )

Performance counter stats for './build/licm\_20\_optimized\_without\_LICM' (10 runs):

936,34 msec task-clock # 0,998 CPUs utilized ( +- 0,34% )

100 context-switches # 0,107 K/sec ( +- 5,33% )

0 cpu-migrations # 0,000 K/sec

2 061 page-faults # 0,002 M/sec

3 631 041 962 cycles # 3,878 GHz ( +- 0,31% )

8 112 528 336 instructions # 2,23 insn per cycle ( +- 0,00% )

102 188 373 branches # 109,136 M/sec ( +- 0,04% )

37 041 branch-misses # 0,04% of all branches ( +- 3,83% )

0,93794 +- 0,00343 seconds time elapsed ( +- 0,37% )

Performance counter stats for './build/licm\_div\_5\_baseline' (10 runs):

1 050,01 msec task-clock # 0,995 CPUs utilized ( +- 0,07% )

124 context-switches # 0,118 K/sec ( +- 15,46% )

0 cpu-migrations # 0,000 K/sec ( +-100,00% )

606 page-faults # 0,577 K/sec

4 061 764 336 cycles # 3,868 GHz ( +- 0,02% )

3 207 038 568 instructions # 0,79 insn per cycle ( +- 0,01% )

201 264 407 branches # 191,678 M/sec ( +- 0,02% )

32 841 branch-misses # 0,02% of all branches ( +- 5,62% )

1,05559 +- 0,00487 seconds time elapsed ( +- 0,46% )

Performance counter stats for './build/licm\_div\_5\_optimized' (10 runs):

211,54 msec task-clock # 0,998 CPUs utilized ( +- 0,46% )

20 context-switches # 0,095 K/sec ( +- 4,03% )

0 cpu-migrations # 0,000 K/sec

606 page-faults # 0,003 M/sec

817 216 088 cycles # 3,863 GHz ( +- 0,04% )

1 505 302 543 instructions # 1,84 insn per cycle ( +- 0,00% )

100 920 887 branches # 477,081 M/sec ( +- 0,01% )

20 234 branch-misses # 0,02% of all branches ( +- 1,77% )

0,21191 +- 0,00105 seconds time elapsed ( +- 0,49% )

Performance counter stats for './build/licm\_div\_5\_optimized\_without\_LICM' (10 runs):

1 042,53 msec task-clock # 0,994 CPUs utilized ( +- 0,08% )

104 context-switches # 0,100 K/sec ( +- 4,90% )

0 cpu-migrations # 0,000 K/sec

606 page-faults # 0,581 K/sec

4 035 980 520 cycles # 3,871 GHz ( +- 0,01% )

3 306 722 238 instructions # 0,82 insn per cycle ( +- 0,00% )

101 211 336 branches # 97,083 M/sec ( +- 0,01% )

30 187 branch-misses # 0,03% of all branches ( +- 2,36% )

1,04855 +- 0,00539 seconds time elapsed ( +- 0,51% )

Performance counter stats for './build/licm\_div\_10\_baseline' (10 runs):

2 088,43 msec task-clock # 0,998 CPUs utilized ( +- 0,05% )

231 context-switches # 0,111 K/sec ( +- 6,33% )

0 cpu-migrations # 0,000 K/sec

1 089 page-faults # 0,521 K/sec

8 102 853 299 cycles # 3,880 GHz ( +- 0,01% )

5 713 295 782 instructions # 0,71 insn per cycle ( +- 0,00% )

202 426 518 branches # 96,928 M/sec ( +- 0,02% )

59 797 branch-misses # 0,03% of all branches ( +- 2,55% )

2,09214 +- 0,00220 seconds time elapsed ( +- 0,11% )

Performance counter stats for './build/licm\_div\_10\_optimized' (10 runs):

379,59 msec task-clock # 0,997 CPUs utilized ( +- 0,30% )

40 context-switches # 0,105 K/sec ( +- 9,32% )

0 cpu-migrations # 0,000 K/sec

1 089 page-faults # 0,003 M/sec

1 469 678 161 cycles # 3,872 GHz ( +- 0,10% )

2 507 491 319 instructions # 1,71 insn per cycle ( +- 0,00% )

101 298 371 branches # 266,862 M/sec ( +- 0,02% )

24 126 branch-misses # 0,02% of all branches ( +- 2,58% )

0,38056 +- 0,00165 seconds time elapsed ( +- 0,43% )

Performance counter stats for './build/licm\_div\_10\_optimized\_without\_LICM' (10 runs):

2 098,93 msec task-clock # 0,999 CPUs utilized ( +- 0,08% )

206 context-switches # 0,098 K/sec ( +- 1,40% )

0 cpu-migrations # 0,000 K/sec

1 089 page-faults # 0,519 K/sec

8 106 793 928 cycles # 3,862 GHz ( +- 0,01% )

5 810 348 170 instructions # 0,72 insn per cycle ( +- 0,00% )

101 881 836 branches # 48,540 M/sec ( +- 0,01% )

41 589 branch-misses # 0,04% of all branches ( +- 1,77% )

2,10189 +- 0,00219 seconds time elapsed ( +- 0,10% )

Performance counter stats for './build/licm\_div\_20\_baseline' (10 runs):

4 179,27 msec task-clock # 0,998 CPUs utilized ( +- 0,13% )

437 context-switches # 0,104 K/sec ( +- 3,16% )

0 cpu-migrations # 0,000 K/sec ( +-100,00% )

2 061 page-faults # 0,493 K/sec

16 148 407 242 cycles # 3,864 GHz ( +- 0,02% )

10 719 234 520 instructions # 0,66 insn per cycle ( +- 0,01% )

203 531 044 branches # 48,700 M/sec ( +- 0,06% )

81 013 branch-misses # 0,04% of all branches ( +- 5,29% )

4,18912 +- 0,00594 seconds time elapsed ( +- 0,14% )

Performance counter stats for './build/licm\_div\_20\_optimized' (10 runs):

836,47 msec task-clock # 0,995 CPUs utilized ( +- 0,43% )

152 context-switches # 0,181 K/sec ( +- 21,78% )

0 cpu-migrations # 0,000 K/sec

2 061 page-faults # 0,002 M/sec

3 201 759 444 cycles # 3,828 GHz ( +- 0,29% )

4 512 150 816 instructions # 1,41 insn per cycle ( +- 0,00% )

102 127 270 branches # 122,093 M/sec ( +- 0,04% )

37 411 branch-misses # 0,04% of all branches ( +- 4,26% )

0,84102 +- 0,00412 seconds time elapsed ( +- 0,49% )

Performance counter stats for './build/licm\_div\_20\_optimized\_without\_LICM' (10 runs):

4 171,73 msec task-clock # 0,998 CPUs utilized ( +- 0,06% )

418 context-switches # 0,100 K/sec ( +- 3,01% )

0 cpu-migrations # 0,000 K/sec

2 061 page-faults # 0,494 K/sec

16 205 060 670 cycles # 3,884 GHz ( +- 0,02% )

10 419 256 240 instructions # 0,64 insn per cycle ( +- 0,01% )

103 534 714 branches # 24,818 M/sec ( +- 0,14% )

76 106 branch-misses # 0,07% of all branches ( +- 6,05% )

4,17898 +- 0,00432 seconds time elapsed ( +- 0,10% )

1. Версия без LICM оптимизации подразумевает, что отключена именно данная оптимизация, в то время как остальные включены. Т.к. для подобного случая нет ключа оптимизации, то для этого в коде LLVM (в файле *llvm-project/llvm/lib/Transforms/Scalar/LICM.cpp*) пришлось вручную отключить LICM-проход компилятора и заново пересобрать LLVM + Clang. В частности, для отключения LICM-прохода:

   После строки №223 (в начало runOnLoop) добавил строку

   return false;

   После строки №350 (в начало LoopInvariantCodeMotion::runOnLoop) добавил строку

   return false; [↑](#footnote-ref-1)