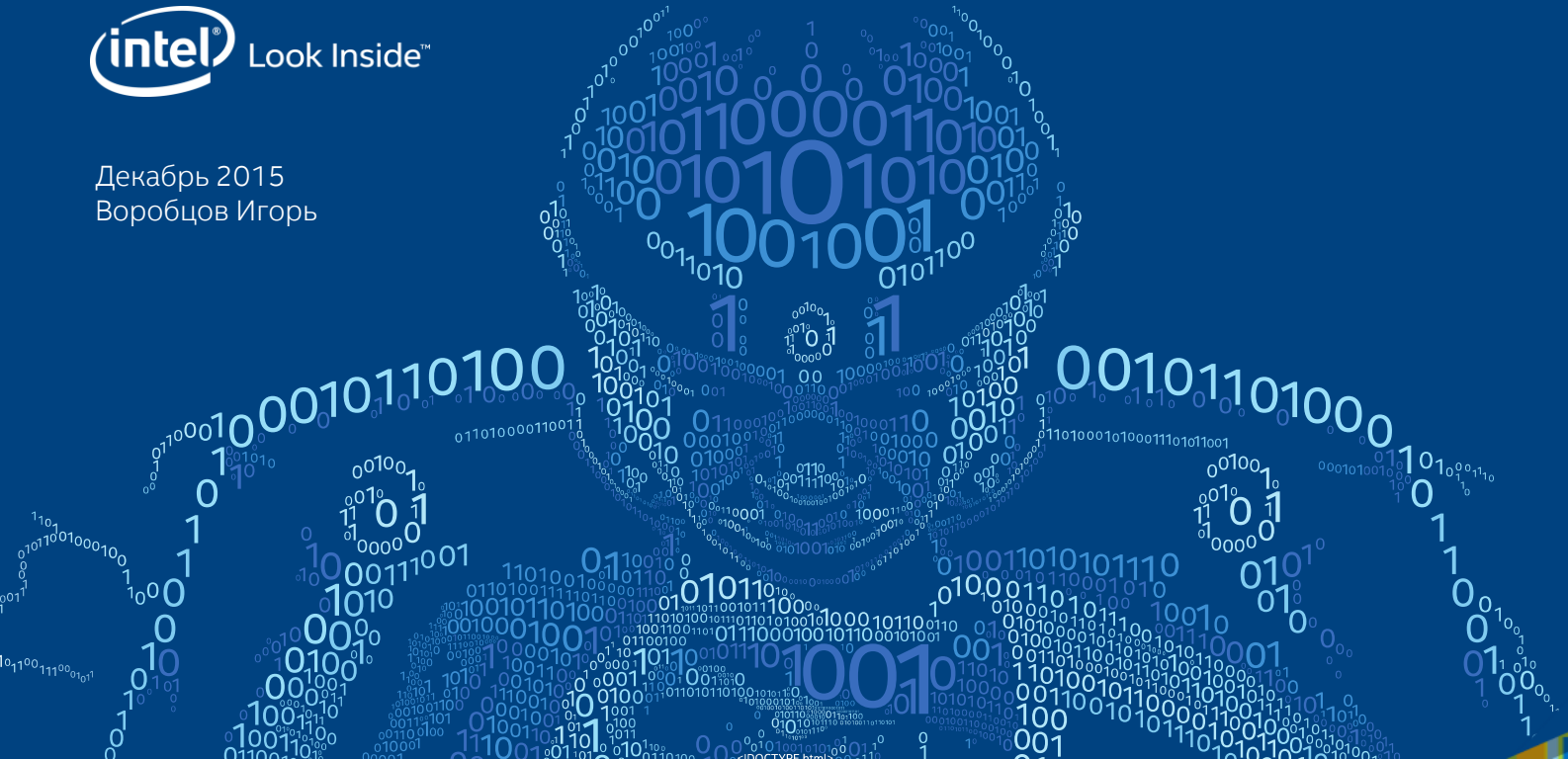
Новые возможности Intel® Parallel Studio XE 2016

intel Software

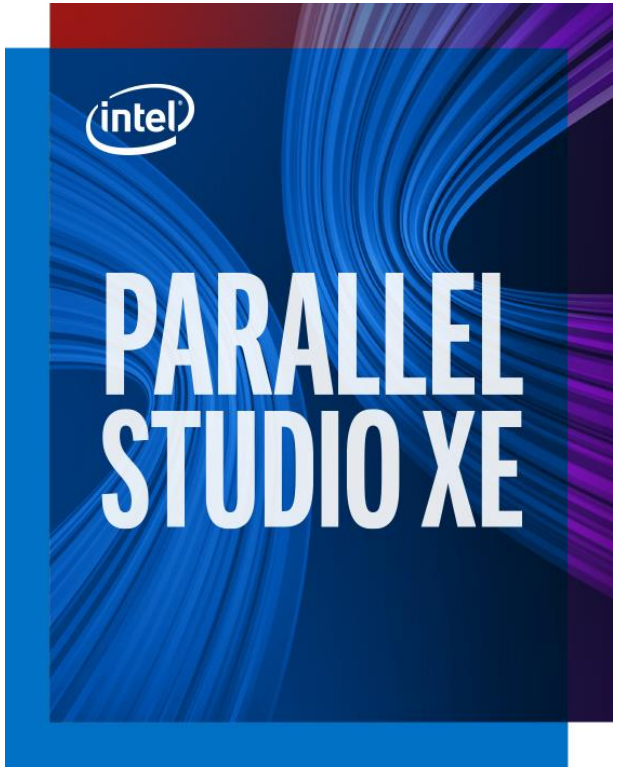
intel Look Inside™

Декабрь 2015
Воробцов Игорь



Intel® Parallel Studio XE 2016

1. Стандарты
2. Аналитика данных
3. Векторизация
4. MPI анализ



Intel® Parallel Studio XE 2016

Много версий хороших и разных

	Composer Edition	Professional Edition	Cluster Edition
Что делает:	***Компилируем производительный код*** с помощью лучших компиляторов и библиотек	***Добавляем средства анализа***	***Добавляем средства по работе с MPI***
Что включает:	• Компиляторы C++ и/или Fortran • Библиотеки • Параллельные модели	Composer edition + • Профилировка • Моделирование параллелизма и работа с векторизацией • Поиск ошибок с памятью и потоками	Professional edition + • Библиотека MPI • Поиск ошибок и оптимизация производительности с MPI

Intel® Parallel Studio XE 2016

Professional Edition

Intel® VTune™ Amplifier XE

Профилировка производительности

Intel® Advisor XE

Моделирование параллелизма и *работа с векторизацией (new)*

Intel® Inspector XE

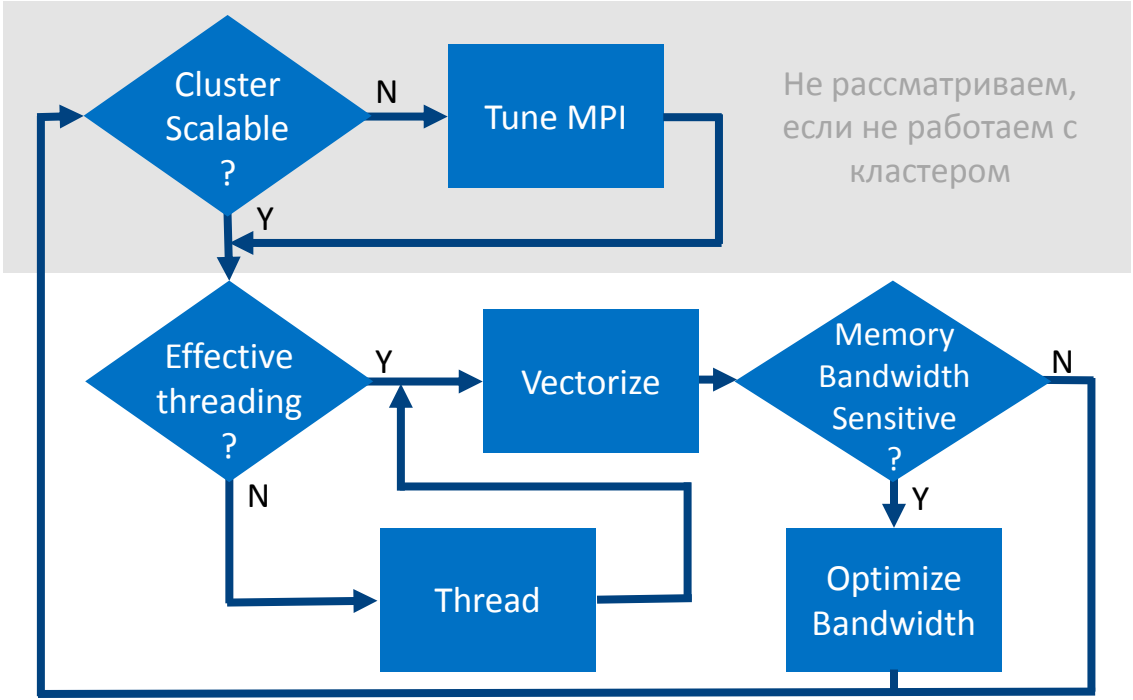
Поиск ошибок

Intel® Parallel Studio XE Composer Edition

Компиляторы и библиотеки

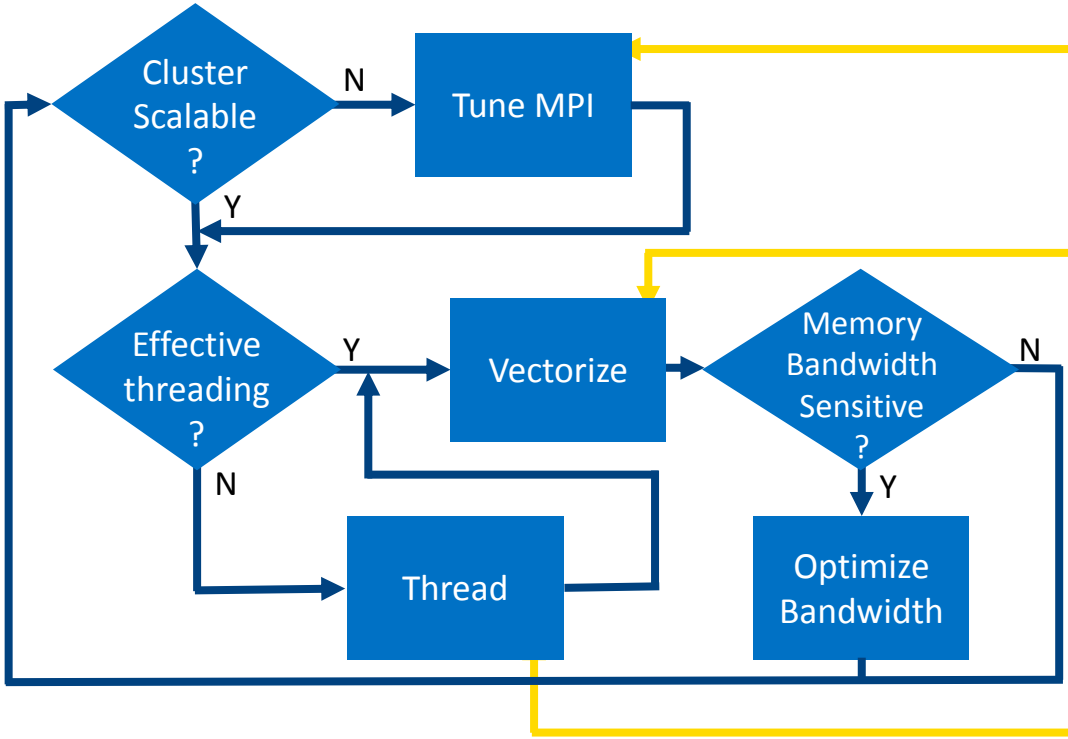
Оптимизация производительности

Итеративный процесс…



Инструменты для реализации

Intel® Parallel Studio XE



Intel® MPI Library
Intel® MPI Benchmarks

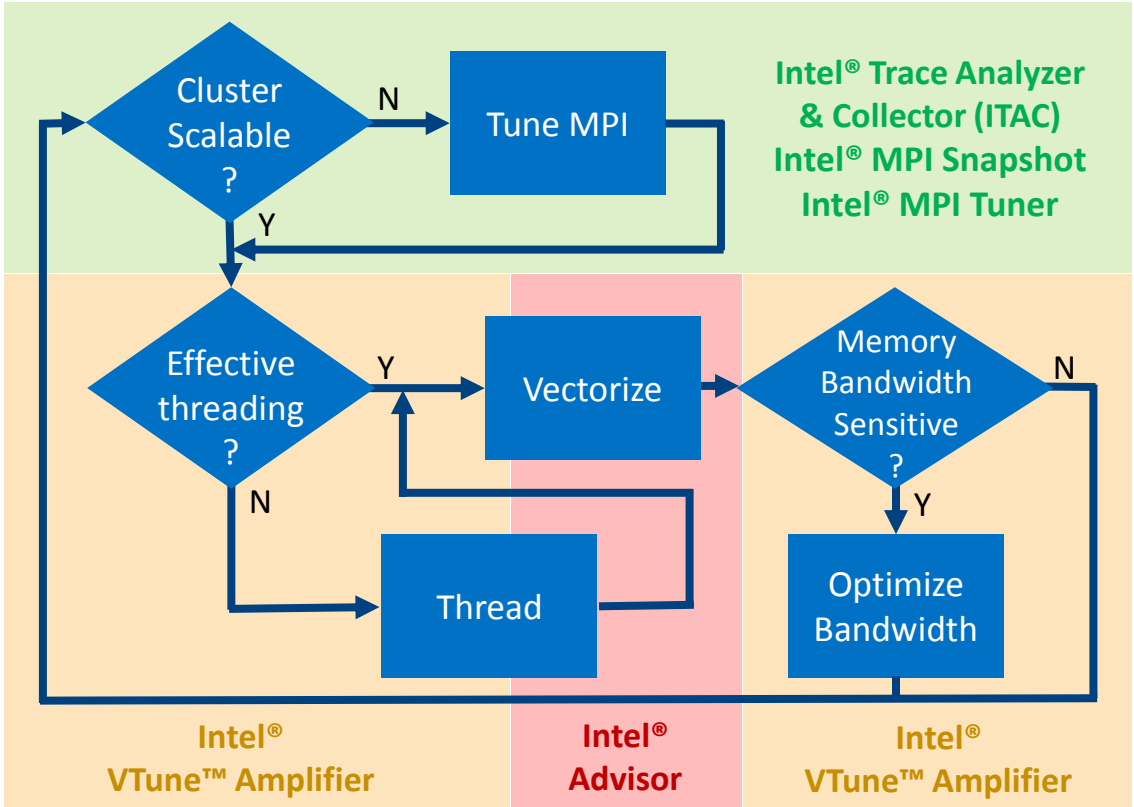
Intel® Compiler

Intel® Math Kernel Library
Intel® IPP – Media & Data Library
Intel® Data Analytics Library
Intel® Cilk™ Plus
Intel® OpenMP*

Intel® TBB – Threading Library

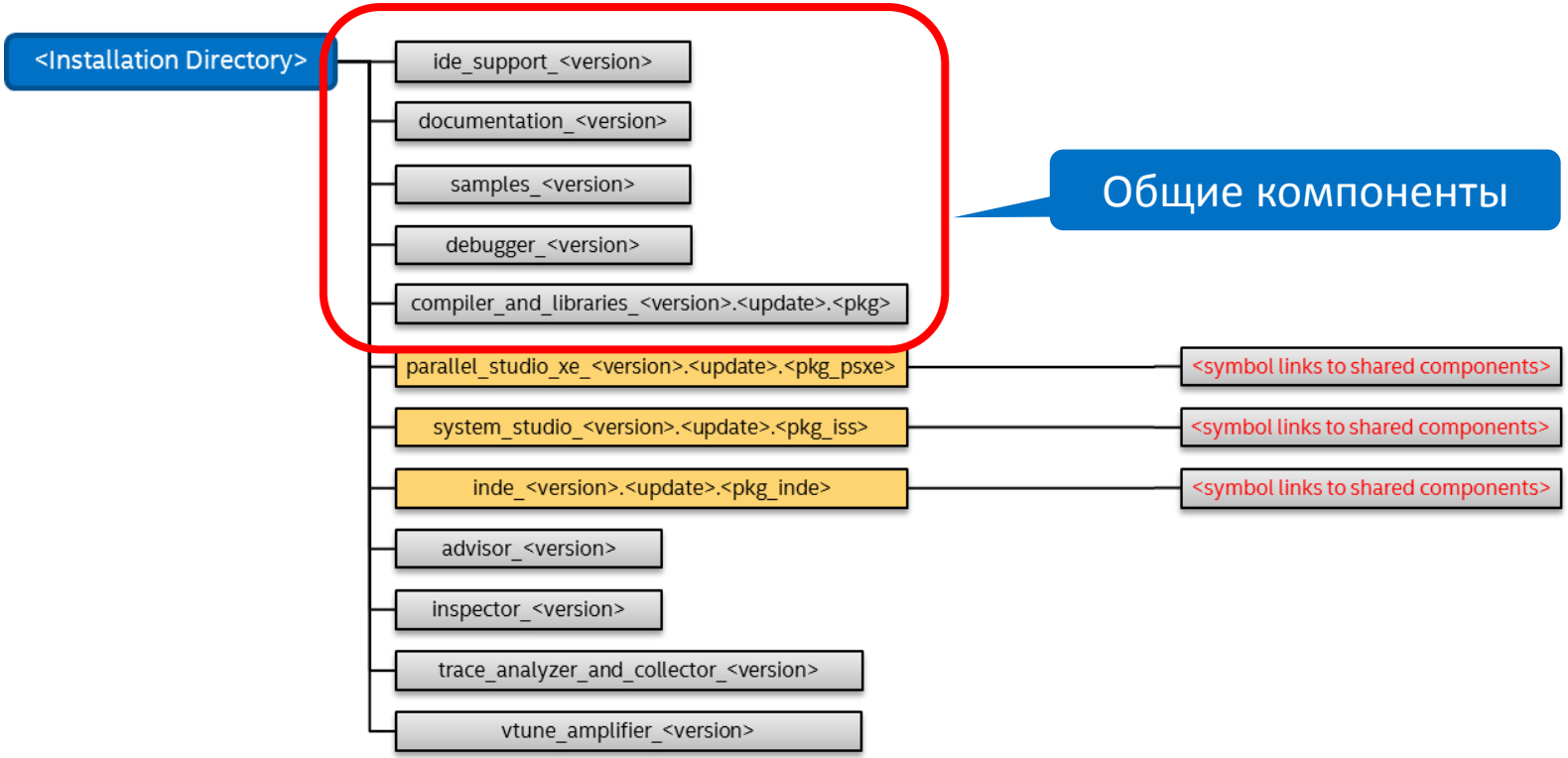
Средства анализа производительности

Intel® Parallel Studio XE



Новая структура директорий

Учим продукты «дружить»



Компиляторы 16.0

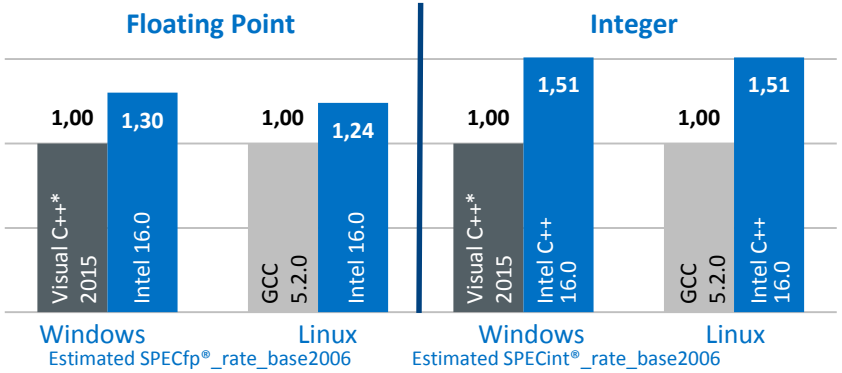
Производительность с последними стандартами

- Больше поддержки C++14
  - C++11 поддерживался полностью ещё в 15.0
- Почти полная поддержка C11
- Расширяем F2008
- Первые шаги в F2015
- OpenMP 4.1 TR3
- Новые возможности оффлоада на Intel® Xeon Phi™

Высокая производительность

Компиляторы Intel® C++ и Fortran

Boost C++ application performance on Windows* & Linux* using Intel® C++ Compiler (higher is better)



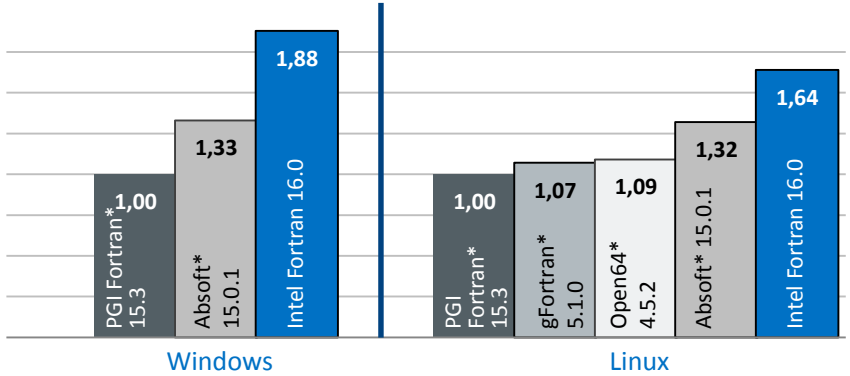
Relative geomean performance, SPEC* benchmark - higher is better

Configuration: Windows hardware: HP DL320e Gen8 v2 (single-socket server) with Intel(R) Xeon(R) CPU E3-1280 v3 @ 3.60GHz, 32 GB RAM, HyperThreading is off; Linux hardware: HP BL460c Gen9 with Intel(R) Xeon(R) CPU E5-2680 v3 @ 2.50GHz, 256 GB RAM, HyperThreading is on. Software: Intel C++ compiler 16.0, Microsoft (R) C/C++ Optimizing Compiler Version 19.00.23026 for x86/x64, GCC 5.2.0. Linux OS: Red Hat Enterprise Linux Server release 7.1 (Maipo), kernel 3.10.0-229.el7.x86_64. Windows OS: Windows 8.1. SPEC* Benchmark ( ).

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. * Other brands and names are the property of their respective owners. Benchmark Source: Intel Corporation

Optimization Notice: Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice. Notice revision #20110804 .

Boost Fortran application performance on Windows* & Linux* using Intel® Fortran Compiler (higher is better)



Relative geomean performance, Polyhedron* benchmark– higher is better

Configuration: Hardware: Intel(R) Core(TM) i7-4770K CPU @ 3.50GHz, HyperThreading is off, 16 GB RAM. Software: Intel Fortran compiler 16.0, Absoft*15.0.1, PGI Fortran* 15.3, Open64* 4.5.2, gFortran* 5.1.0. Linux OS: Red Hat Enterprise Linux Server release 7.0 (Maipo), kernel 3.10.0-123.el7.x86_64. Windows OS: Windows 7, Service pack 1. Polyhedron Fortran Benchmark ( ). Windows compiler switches: Absoft: -m64 -O5 -speed_math=10 -fast_math -march=core -xINTEGER -stack:0x80000000. Intel* Fortran compiler: /fast /Qparallel /link /stack:64000000. PGI Fortran: -fastsse -Munroll=n:4 -Mipa=fast,inline -Mconcur=numa.
Linux compiler switches: Absoft -m64 -mavx -O5 -speed_math=10 -march=core -xINTEGER, Gfortran: -Ofast -mfpmath=sse -flto -march=native -funroll-loops -ftree-parallelize-loops=4. Intel Fortran compiler: -fast –parallel. PGI Fortran: -fast -Mipa=fast,inline -Msmartalloc -Mfprelaxed -Mstack_arrays -Mconcur=bind. Open64: -march=bdver1 -mavx -mno-fma4 -Ofast -mso –apo.

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. * Other brands and names are the property of their respective owners. Benchmark Source: Intel Corporation

Optimization Notice: Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice. Notice revision #20110804 .

C++14

Производительность с последними стандартами

- */Qstd=c++14* на Windows и *-std=c++14* на Linux и Mac OS X
- https://software.intel.com/en-us/articles/c14-features-supported-by-intel-c-compiler
  - Обобщённые лямбда-функции
  - Захват выражений для лямбда-функций
  - Разделители разрядов
  - Атрибут [[deprecated]]
  - Вывод типа возвращаемого значения для функций
  - Агрегатная инициализация классов с инициализаторами полей

C++14 TR

Feature Test макрос

```
#if __cpp_binary_literals
int const packed_zero_to_three = 0b00011011;
#else
int const packed_zero_to_three = 0x1B;
#endif
```

```
#if __has_include("shared_mutex")  // use standard header
#elif __has_include("boost/shared_mutex.h") // use BOOST header
#endif
```

C11

Поддержка стандарта C

- */Qstd=c11* на Windows и *-std=c11* на Linux и Mac OS X
- https://software.intel.com/en-us/articles/c11-support-in-intel-c-compiler

`_Alignas`	`_Alignof`
`_Static_assert`	`_Thread_local`
`_Noreturn`	`_Generic`

C11

Выравнивание

- Раньше решения были зависимы от компилятора

```
__declspec(align(base)) <var>
<var> __attribute__((aligned(base)))
```

```
#ifdef __GNUC__
#define _ALIGN(N)   __attribute__((aligned(N)))
#else
#define _ALIGN(N)   __declspec(align(N))
#endif

_ALIGN(16) int foo[4];
```

C11

Выравнивание

- Раньше решения были зависимы от компилятора

```
__declspec(align(base)) <var>
<var> __attribute__((aligned(base)))
```

```
// массив  cacheline выравнен по 64 байта
_Alignas(64) char cacheline[64];
printf("Alignment of char = %d\n", _Alignof(char));
```

C11
«Шаблоны для C»

- Выражения, не зависящие от типа, с использованием ключевого слова _Generic

```
#define sqrt(x) _Generic((x), long double: sqrtl, \
                              default: sqrt, \
                              float: sqrtf)(x)
```

C11

Анонимные структуры и объединения

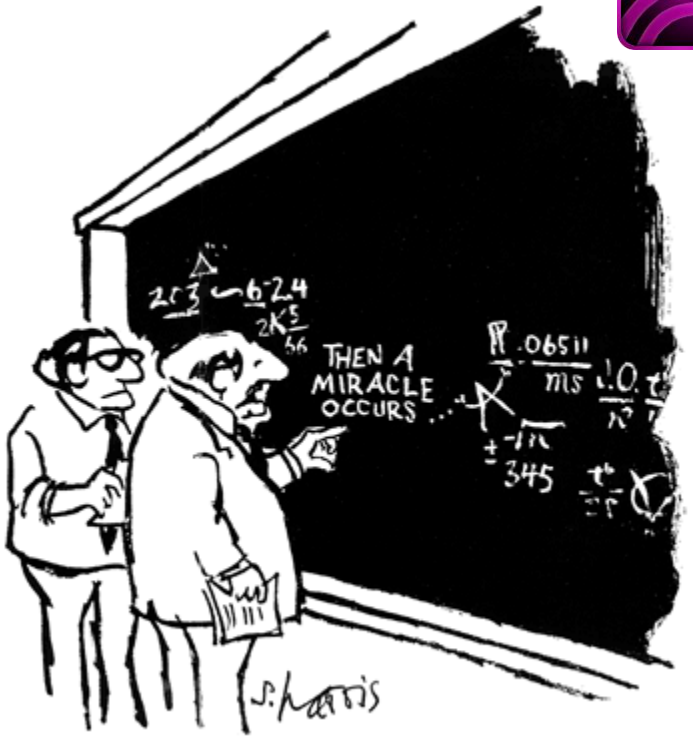
- Для вложения структур и объединений

```
struct T // C11
{
  int m;
  union //анонимное объединение
  {
    char * index;
    int key;
  };
};
struct T t;
t.key = 1300; //прямой доступ к члену объединения key
```

Приоритет скобок

- *A+(B+C)*
  - компилятор не обязан соблюдать приоритет для вычислений
- *-fprotect-parens* (Linux* OS и OS X*) или

  /Qprotect-parens (Windows*)
- `fp-model`



Оптимизация циклов

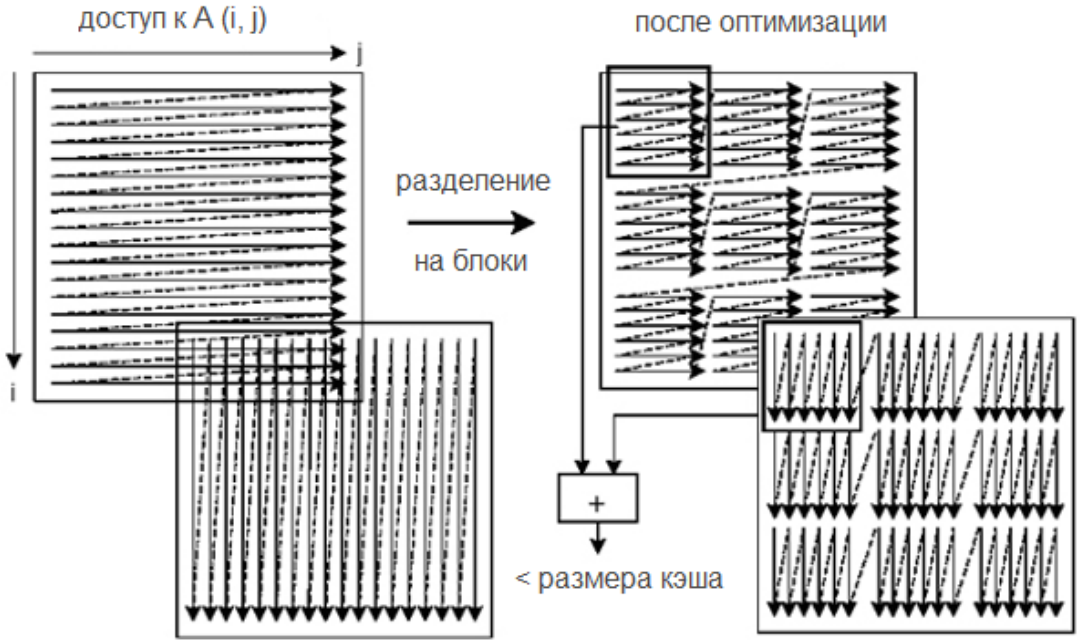
Loop blocking

```
double A[MAX, MAX], B[MAX, MAX];
for (i=0; i< MAX; i++)
  for (j = 0; j< MAX; j++)
    A[i, j] = A[i, j] + B[j, i];
```

```
for (i = 0; i< MAX; i += block_size)
  for (j = 0; j< MAX; j += block_size)
    for (ii = i; ii<i + block_size; ii++)
      for (jj = j; jj<j + block_size; jj++)
        A[ii, jj] = A[ii, jj] + B[jj, ii];
```

Оптимизация циклов

Loop blocking



Оптимизация циклов

Директива block_loop

C++:

```
#pragma block_loop [clause]
#pragma noblock_loop
```

Fortran:

```
!DIR$ BLOCK_LOOP [clause]
!DIR$ NOBLOCK_LOOP
```

Уровень оптимизации *O3*

```
#pragma block_loop factor(256) level(1:2)
for (j = 1; j<n; j++){
  f = 0;
  for (i = 1; i<n; i++){
    f = f + a[i] * b[i];
  }
  c[j] = c[j] + f;
}
```

Оптимизация циклов

Директива block_loop

C++:

```
#pragma block_loop [clause]
#pragma noblock_loop
```

Fortran:

```
!DIR$ BLOCK_LOOP [clause]
!DIR$ NOBLOCK_LOOP
```

```
for (jj = 1; jj<n / 256 + 1; jj + ){
  for (ii = 1; ii<n / 256 + 1; ii++){
    for (j = (jj - 1) * 256 + 1; j<min(jj * 256, n); j++){
      f = 0;
      for (i = (ii - 1) * 256 + 1; i<min(ii * 256, n); i++){
        f = f + a[i] * b[i];
      }
    c[j] = c[j] + f;
    }
  }
}
```

Подмодули из F2008

Проблема

```
module bigmod
…
contains
subroutine sub1
…
function func2
…
subroutine sub47
…
end module bigmod
```

```
! Source source1.f90
…
Call sub1
```

```
! Source source2.f90
…
x = func2(…)
```

```
! Source
source47.f90
…
call sub47
```

Подмодули из F2008

Проблема

```
module bigmod
…
contains
subroutine sub1
…
function func2
…
subroutine sub47
edit

…
end module bigmod
```

```
! Source source1.f90
…
Call sub1
```

```
! Source source2.f90
…
x = func2(…)
```

```
! Source
source47.f90
…
call sub47
```

Подмодули из F2008

Проблема

```
module bigmod
…
contains
subroutine sub1
…
function func2
…
subroutine sub47
edit

…
end module bigmod
```

Компиляция!

```
! Source source1.f90
…
Call sub1
```

Компиляция!

```
! Source source2.f90
…
x = func2(…)
```

Компиляция!

```
! Source
source47.f90
…
call sub47
```

Компиляция!

Подмодули из F2008

Решение

```
module bigmod
…
interface
module subroutine sub1
…
module function func2
…
module subroutine sub47
…
end interface
end module bigmod
```

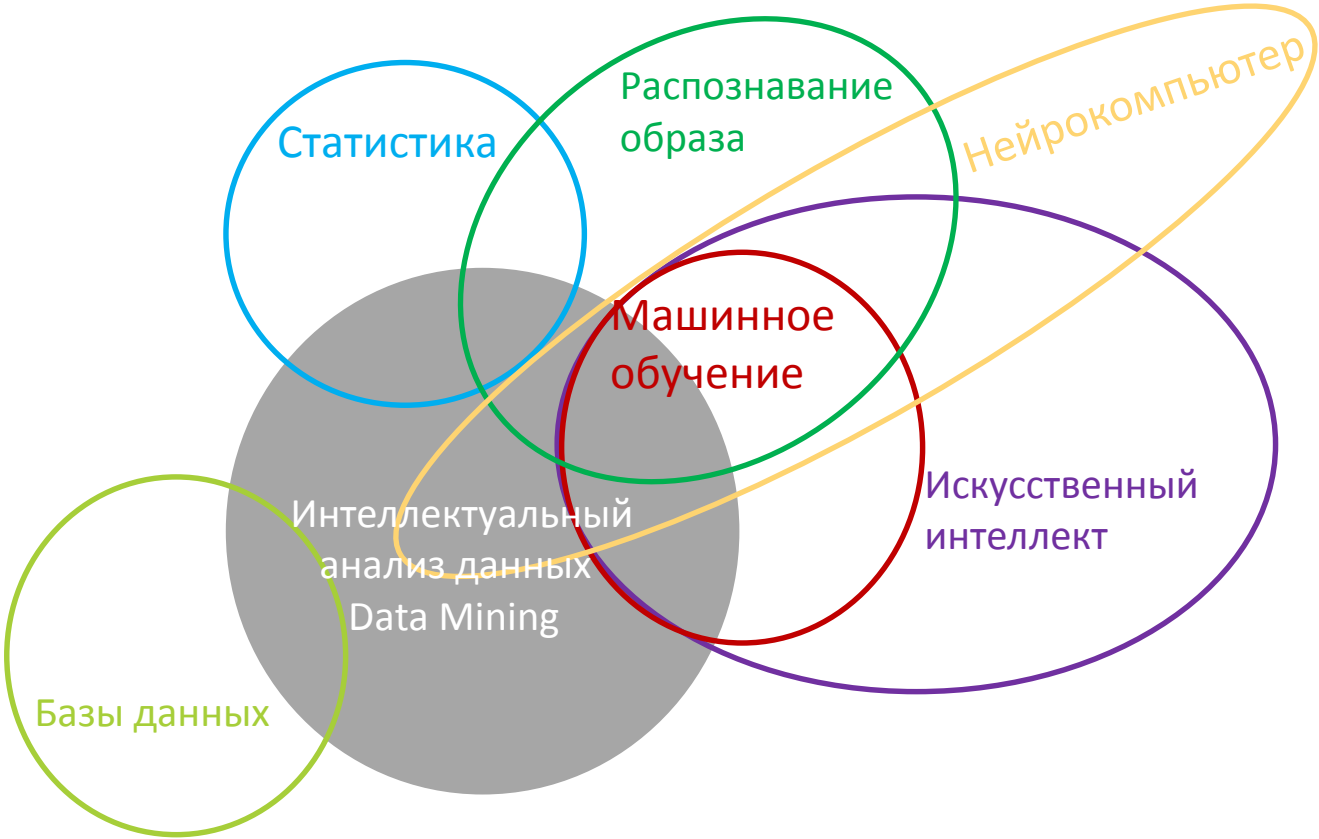
```
submodule (bigmod) bigmod_submod
contains

module subroutine sub1
… implementation of sub1

module function func2
… implementation of func2
…
module subroutine sub3
… implementation of sub3

end submodule bigmod_submod
```

Интероперабельность C и Фортрана
F2015

- *TS29113 (Further Interoperability of Fortran with C)*
  - передача в C функции указателей (pointer), перенимающих конфигурацию (assumed-shape) и выделяемых (allocatable) переменных
  - интероперабельность с C типом void *

- *TYPE(*)*
- *DIMENSION(..)*
- *C descriptor*

Библиотека Intel® Data Analytics Acceleration Library 2016



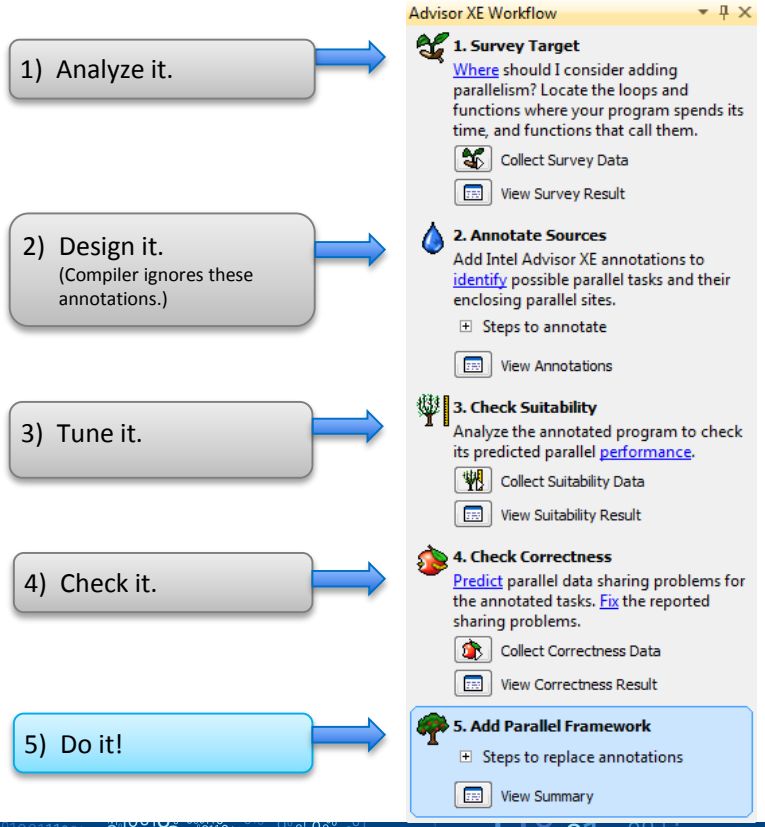
Intel® DAAL

CRISP-DM Cross Industry Standard Process for Data Mining

- Итеративный процесс
- DAAL фокусирован на этапах подготовки данных, моделирования и оценки результатов

Intel® Advisor XE

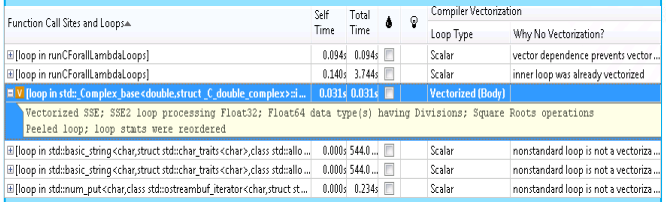
Моделирование параллелизма по задачам



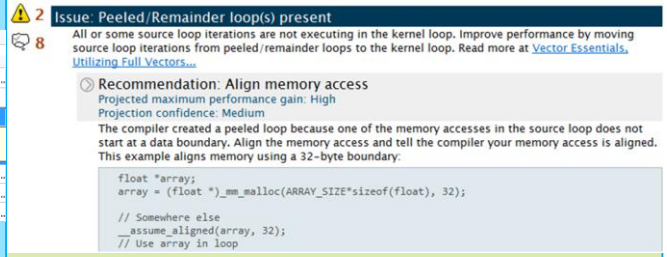
Intel® Advisor XE

Векторизация

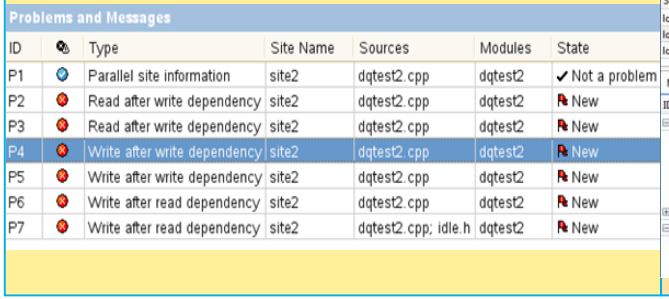
1. Диагностика SIMD циклов



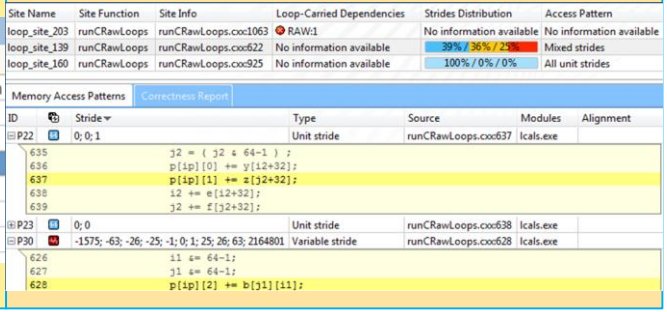
2. Рекомендации



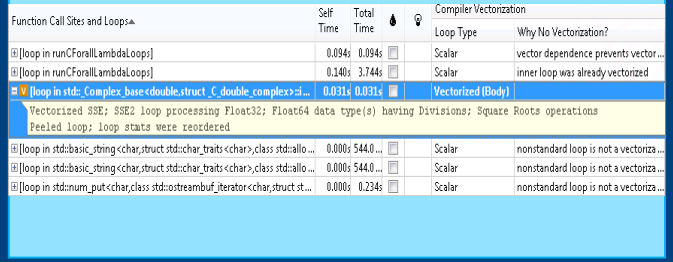
3. Анализ зависимостей



4. Анализ доступов к памяти

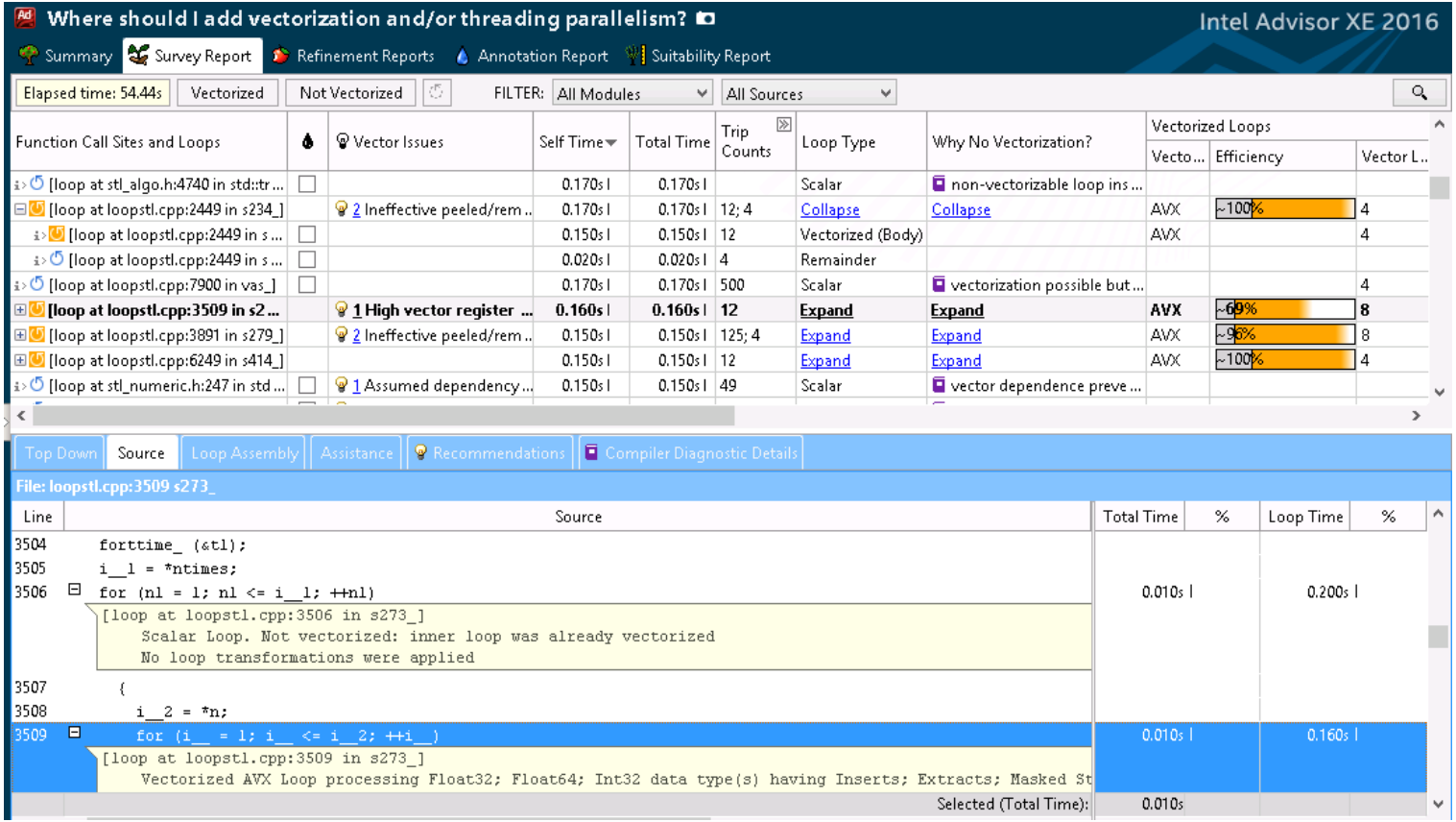


1. Диагностика SIMD

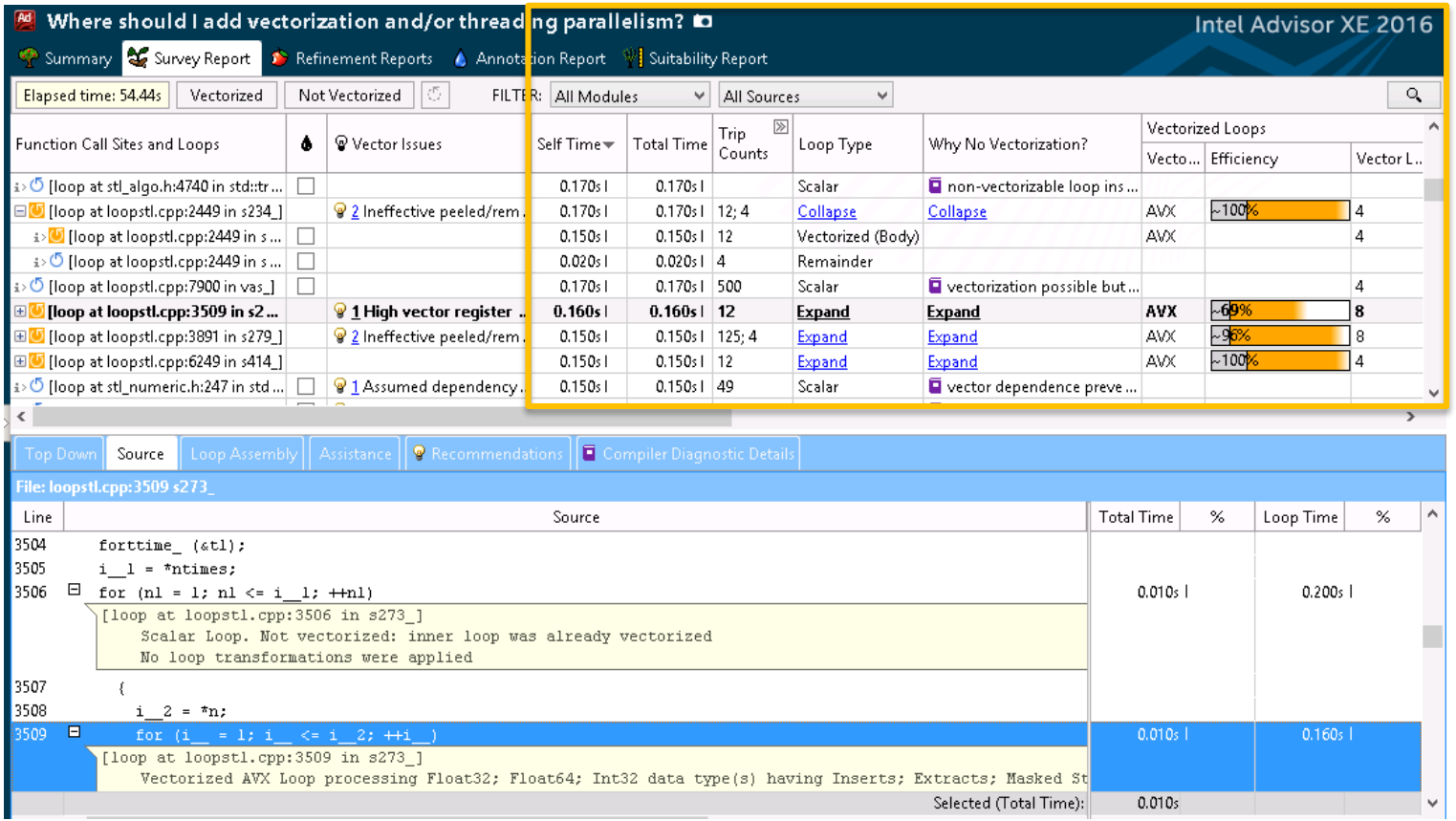


Function Call Sites and Loops	Self Time	Total Time	Compiler Vectorization	
			Loop Type	Why No Vectorization?
[loop in runCForallLambdaLoops]	0.094s	0.094s	Scalar	vector dependence prevents vector...
[loop in runCForallLambdaLoops]	0.140s	3.744s	Scalar	inner loop was already vectorized
[loop in std::_Complex_base<double,struct _C_double_complex>::i...	0.031s	0.031s	Vectorized (Body)	
Vectorized SSE; SSE2 loop processing Float32; Float64 data type(s) having Divisions; Square Roots operations Peeled loop; loop stmts were reordered				
[loop in std::basic_string<char,struct std::char_traits<char>,class std::allo...	0.000s	544.0...	Scalar	nonstandard loop is not a vectoriza...
[loop in std::basic_string<char,struct std::char_traits<char>,class std::allo...	0.000s	544.0...	Scalar	nonstandard loop is not a vectoriza...
[loop in std::num_put<char,class std::ostreambuf_iterator<char,struct st...	0.000s	0.234s	Scalar	nonstandard loop is not a vectoriza...

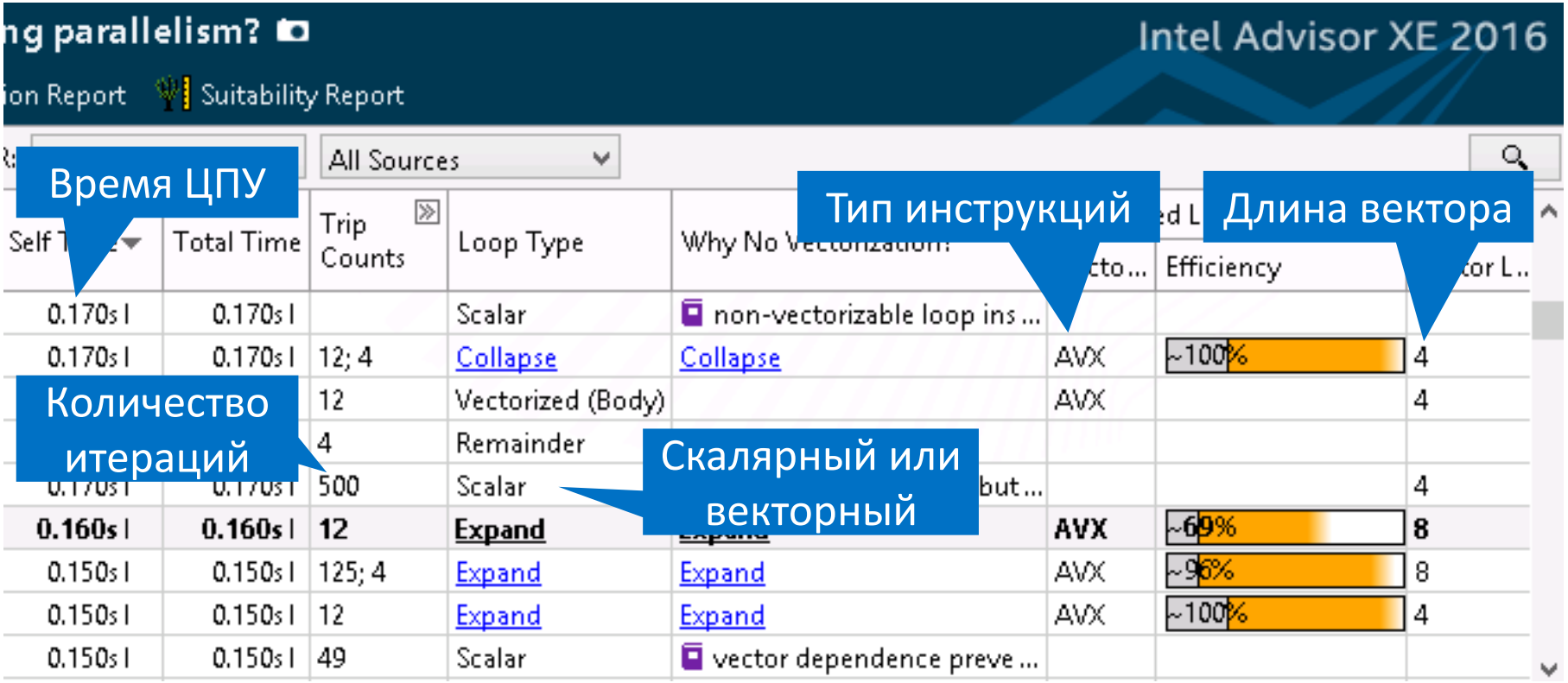
Диагностика SIMD циклов



Диагностика SIMD циклов



Диагностика SIMD циклов



Диагностики в исходном коде

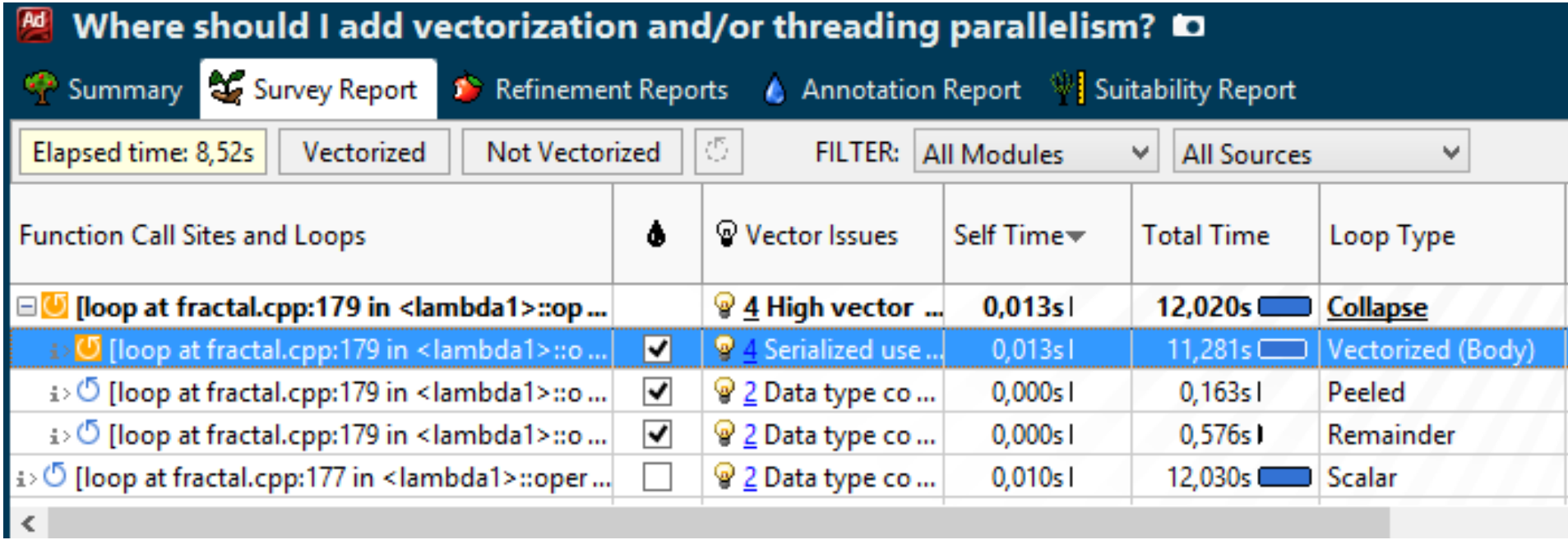
File: fractal.cpp:164 <lambda1>::operator()

Line	Source	Total Time	%
163	for (int x = x0; x < x1; ++x) {		
	[loop at fractal.cpp:163 in <lambda1>::operator()] Scalar Loop. Not vectorized: outer loop was not auto-vectorized: consider usi No loop transformations were applied		
164	for (int y = y0; y < y1; ++y) {		
	[loop at fractal.cpp:164 in <lambda1>::operator()] Scalar Loop. Not vectorized: vectorization possible but seems inefficient. Us Loop was unrolled by 2		
165	fractal_data_array[x - x0][y - y0] = calc_one_pixel(x, y, t	10.822s	
166	}		
167	}		
168	for (int y = y0, y_temp = 0; y < y1; ++y, ++y_temp) {		
169	area.set_pos(0, y - y0);		
170	for (int x = x0, x_temp = 0; x < x1; ++x, ++x_temp) {		
171	area.put_pixel(fractal_data_array[x_temp][y_temp]);		
172	}		
173	}	0.196s	

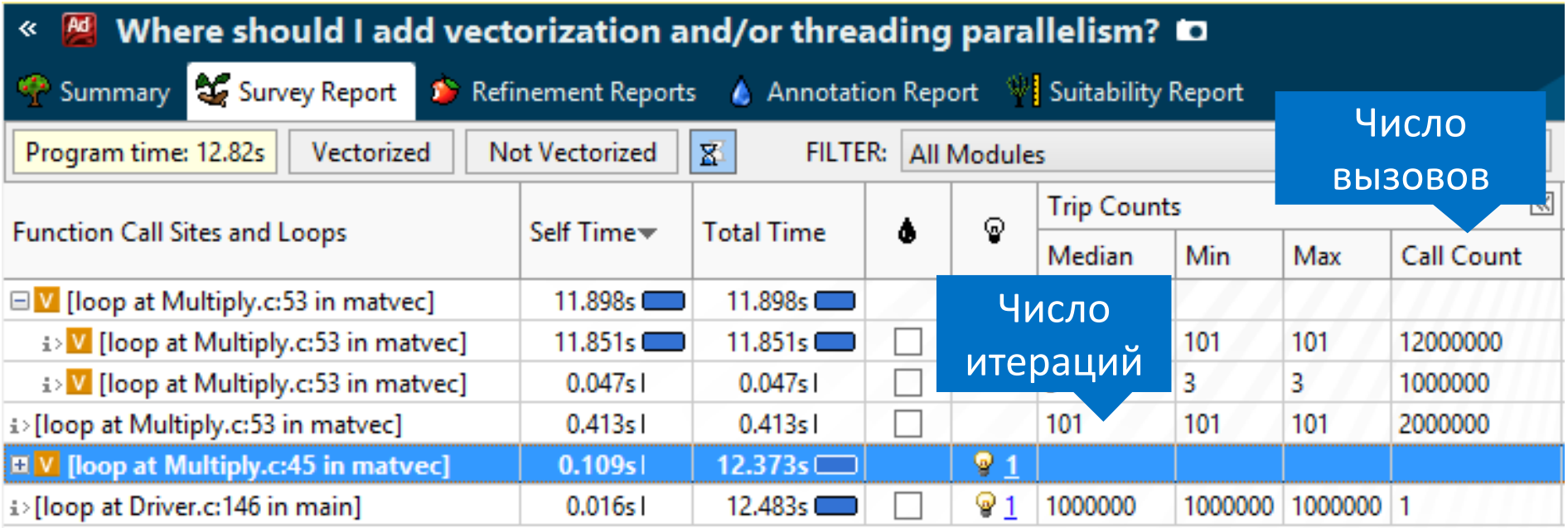
Эффективность векторного цикла

Loops	Vecto...	Efficiency	Estimated Gain	Vect...
[loop at lbpSUB.cpp:1280 in fPropagationS ...	AVX	13%	0,53	4
[loop at lbpGET.cpp:152 in fGetFracSite]	AVX	30%	2,38	8
[loop at lbpGET.cpp:42 in fGetOneMassSite]	AVX	36%	2,86	8
[loop at lbpGET.cpp:78 in fGetTotMassSite]	AVX	36%	2,86	8
[loop at lbpGET.cpp:334 in fGetOneDirecSp ...	AVX	38%	3,05	8
[loop at lbpBGK.cpp:840 in fCollisionBGK]	AVX	100%	2,05	2

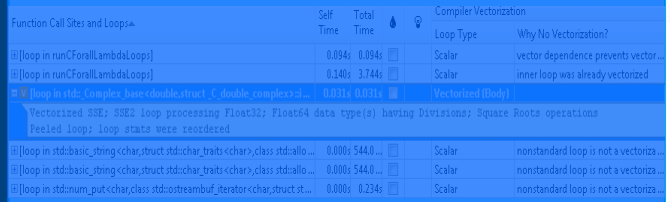
Части векторного цикла



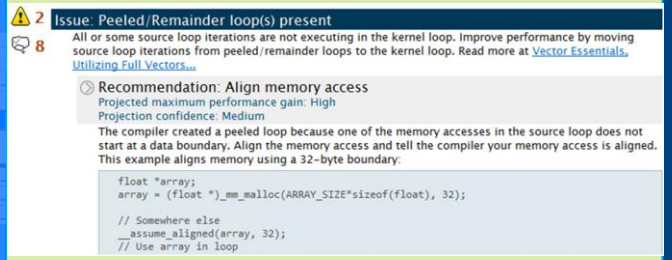
Число итераций



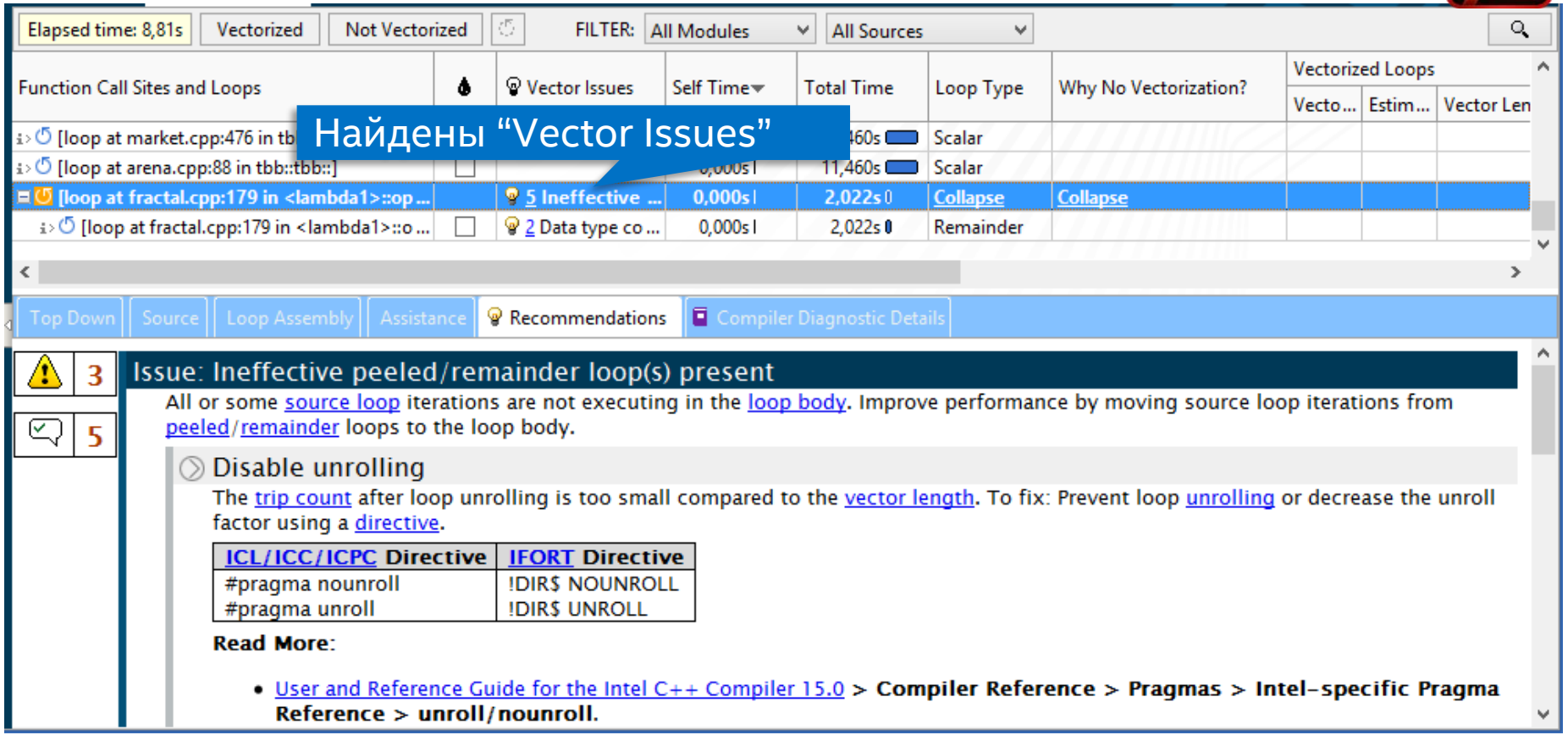
1. Диагностика SIMD



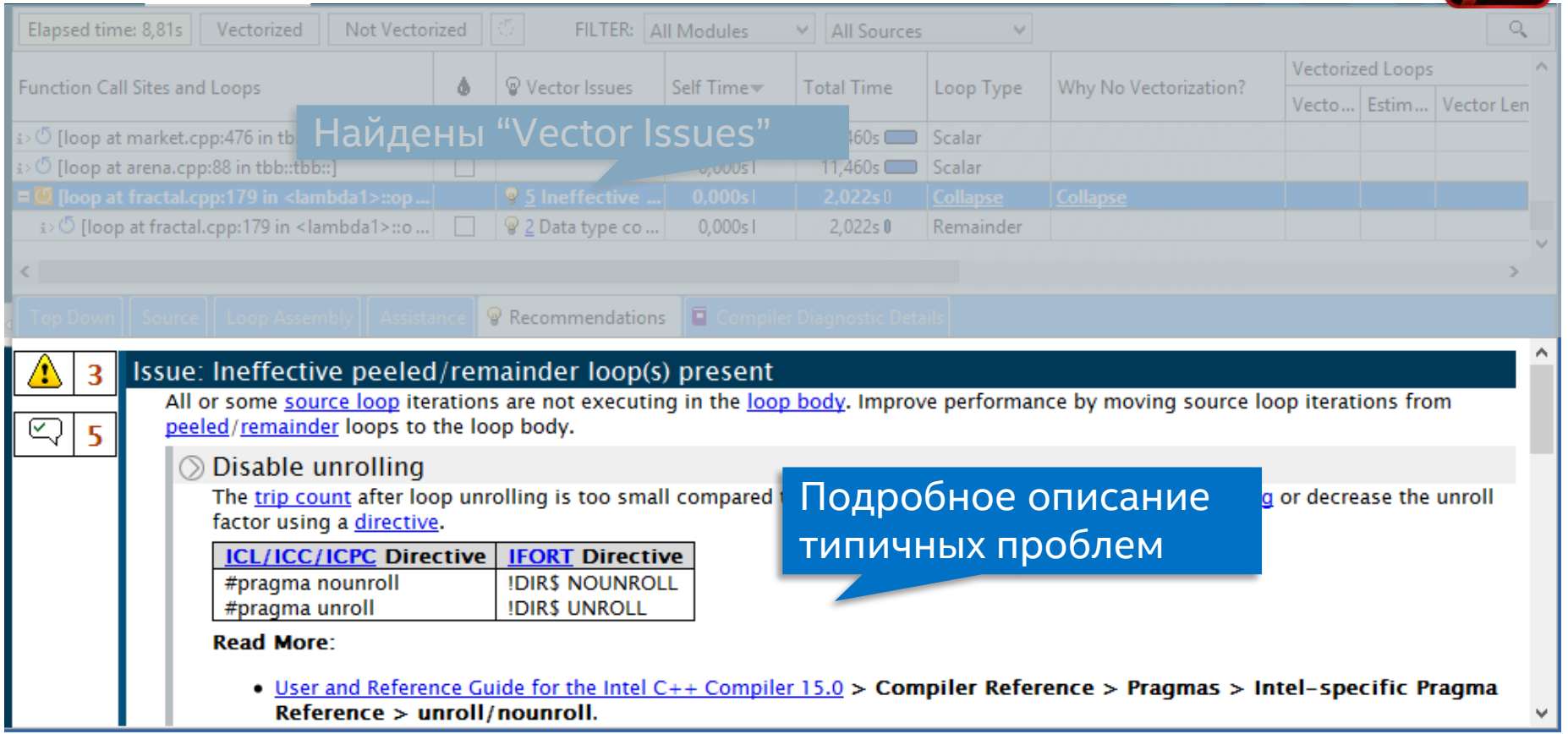
2. Рекомендации



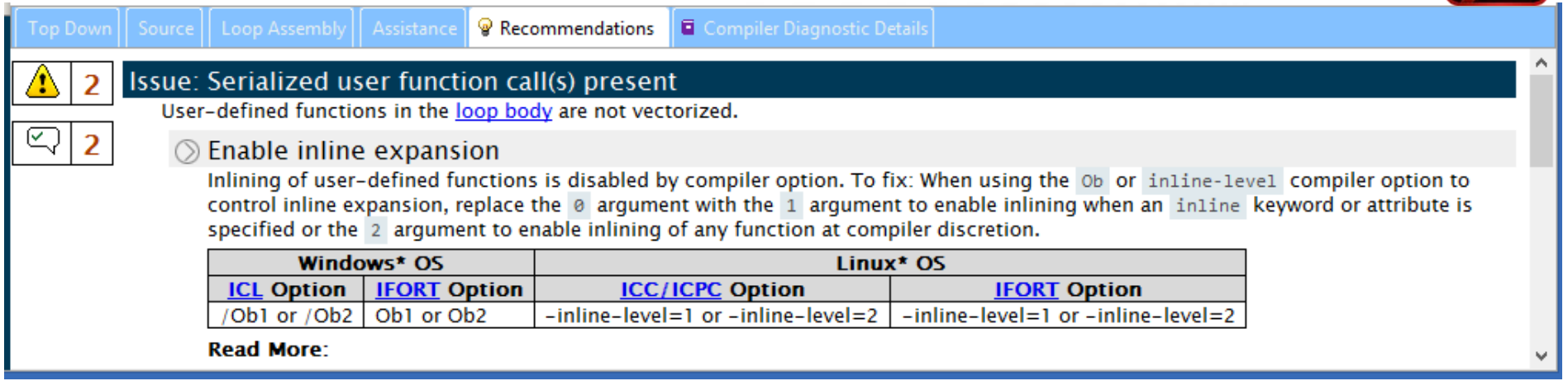
Рекомендации



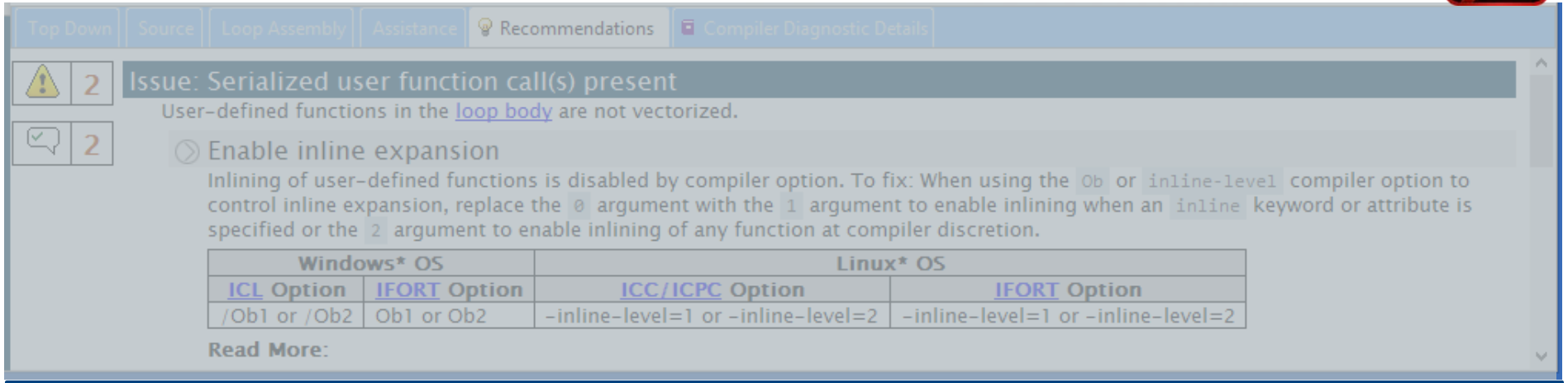
Рекомендации

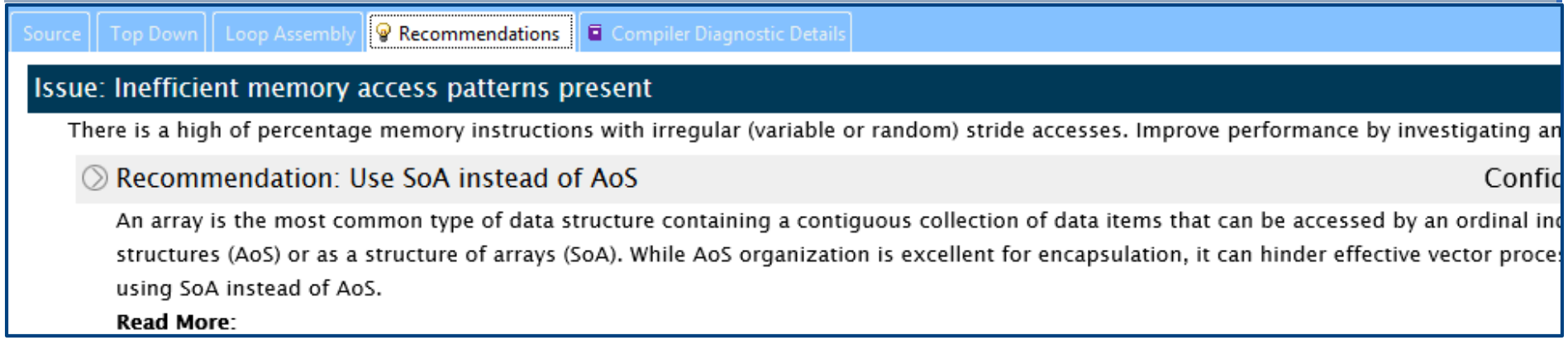


Рекомендации

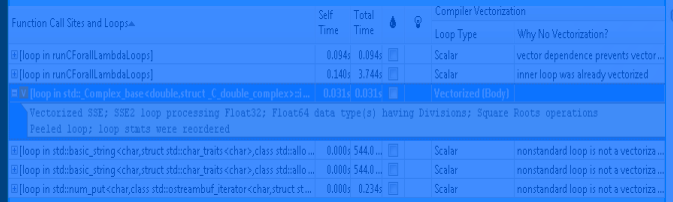


Рекомендации

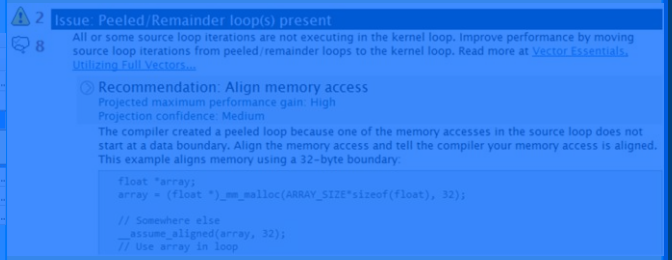




1. Диагностика SIMD циклов



2. Рекомендации



3. Анализ зависимостей

Problems and Messages

ID		Type	Site Name	Sources	Modules	State
P1		Parallel site information	site2	dqtest2.cpp	dqtest2	✔ Not a problem
P2		Read after write dependency	site2	dqtest2.cpp	dqtest2	New
P3		Read after write dependency	site2	dqtest2.cpp	dqtest2	New
P4		Write after write dependency	site2	dqtest2.cpp	dqtest2	New
P5		Write after write dependency	site2	dqtest2.cpp	dqtest2	New
P6		Write after read dependency	site2	dqtest2.cpp	dqtest2	New
P7		Write after read dependency	site2	dqtest2.cpp; idle.h	dqtest2	New

Зависимости по данным

```
DO I = 1, N
   A(I) = A(I-1) * B(I)
ENDDO
```

```
void scale(int *a, int *b)
{
   for (int i = 0; i < 1000; i++)
        b[i] = z * a[i];
}
```

Issue: Assumed dependency present

The compiler assumed there is an anti-dependency (Write after read – WAR) or true dependency (Read after write – RAW) in the loop. Improve performance by investigating the assumption and handling accordingly.

Enable vectorization

Potential performance gain: Information not available until Beta Update release

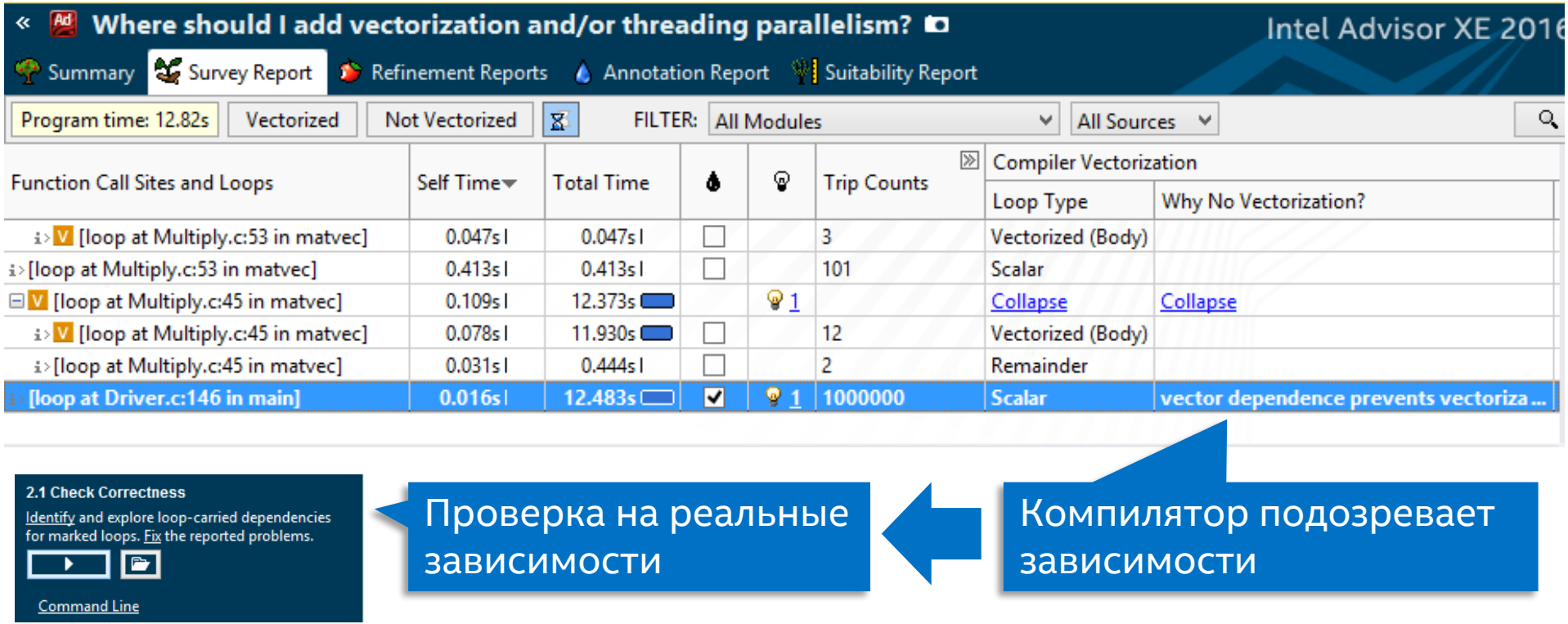
Confidence this recommendation applies to your code: Information not available until Beta Update release

The Correctness analysis shows there is no real dependency in the loop for the given workload. Tell the compiler it is safe to vectorize using the `restrict` keyword or a directive.

ICL/ICC/ICPC Directive	IFORT Directive	Outcome
#pragma simd or #pragma omp simd	!DIR$ SIMD or !$OMP SIMD	Ignores all dependencies in the loop
#pragma ivdep	!DIR$ IVDEP	Ignores only vector dependencies (which is safest)

Read More:

Анализ зависимостей



Проверка на реальные зависимости

Компилятор подозревает зависимости

Реально существующие зависимости

Memory Access Patterns Report | Correctness Report

Problems and Messages

ID	Type	Site Name	Sources	Modules	State
P1	Parallel site information	loop_site_6	main.cpp	test_1.exe	✔ Not a problem
P3	Read after write dependency	loop_site_6	crtexe.c; main.cpp	test_1.exe	New
P4	Write after write dependency	loop_site_6	crtexe.c; main.cpp	test_1.exe	New
P5	Write after read dependency	loop_site_6	crtexe.c; main.cpp	test_1.exe	New

Write after read dependency: Code Locations

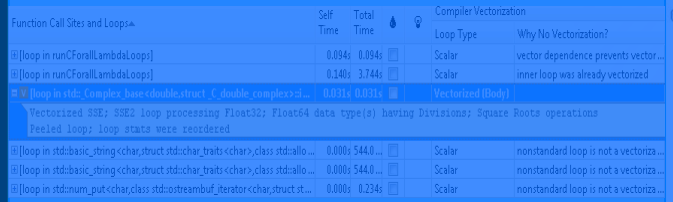
ID	Description	Source	Function	Module	State
X17	Read	main.cpp:22	main	test_1.exe	New

```
20          k += a[9];
21          k *= a[8];
22          k -= a[7];
23          k += a[6];
24          k *= a[5];
```

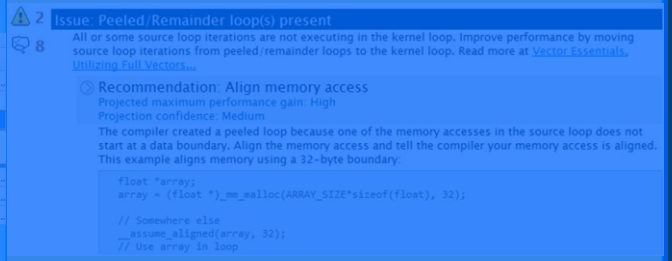
X18	Read	main.cpp:23	main	test_1.exe	New

```
21          k *= a[8];
22          k -= a[7];
23          k += a[6];
```

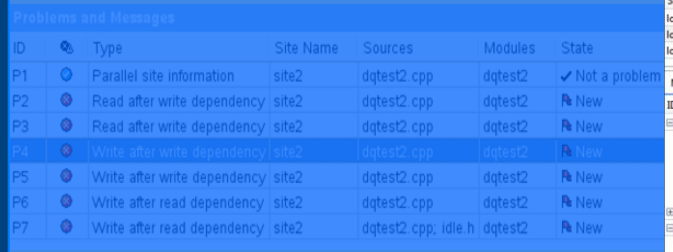
1. Диагностика SIMD циклов



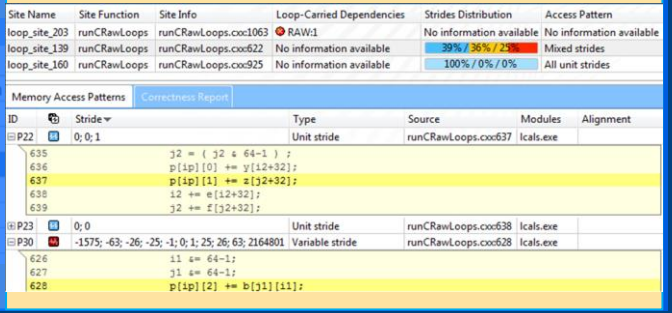
2. Рекомендации



3. Анализ зависимостей



4. Анализ доступов к памяти



Шаблоны доступа к памяти

Unit-Stride access

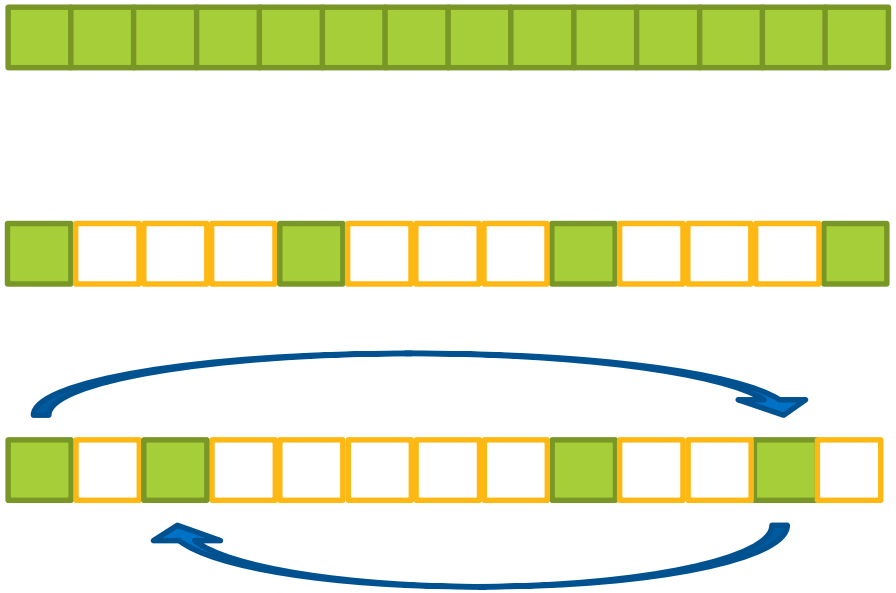
```
for (i=0; i<N; i++)
    A[i] = C[i]*D[i]
```

Constant stride access

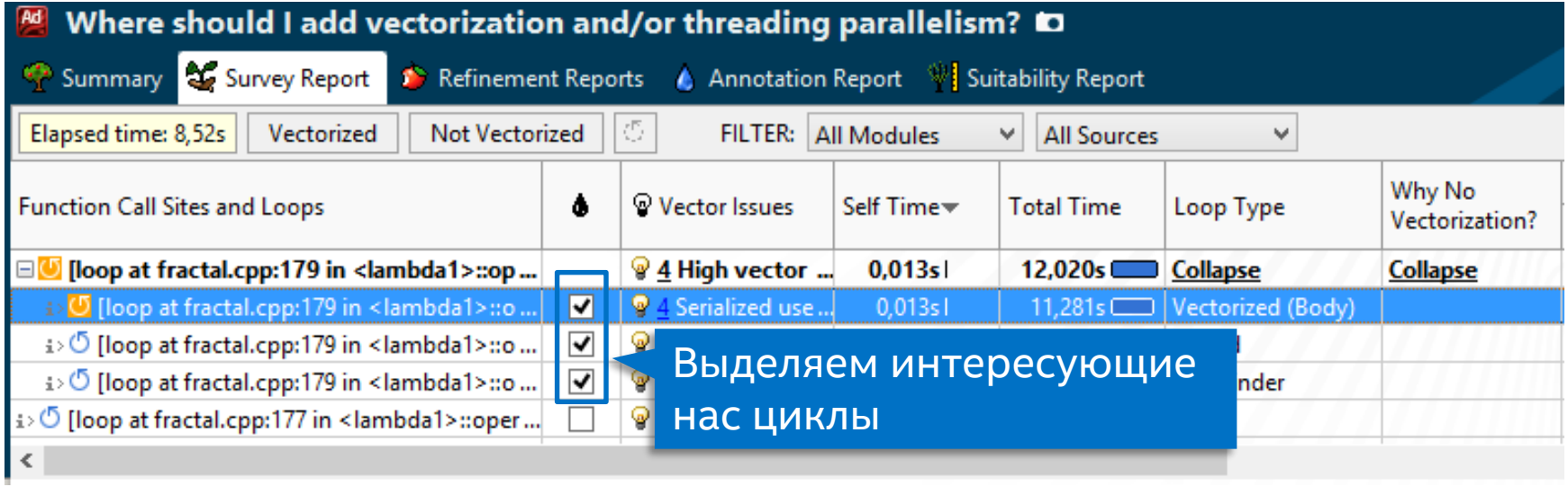
```
for (i=0; i<N; i++)
    point[i].x = x[i]
```

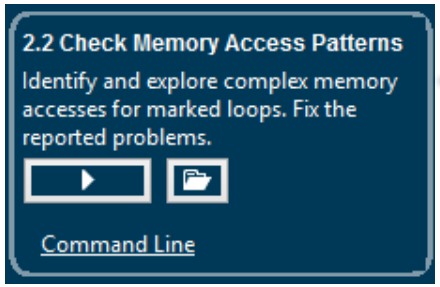
Variable stride access

```
for (i=0; i<N; i++)
    A[B[i]] = C[i]*D[i]
```



Анализ шаблонов доступа





Запускаем анализ шаблонов доступа

Шаблоны доступа

Unit/Constant/Variable

Site Name	Site Function	Site Info	Loop-Carried Dependencies	Strides Distribution	Access Pattern
loop_site_133	grid_intersect	grid.cpp:559	No information available	23% / 1% / 76%	Mixed strides
loop_site_131	grid_intersect	grid.cpp:581	No information available	21% / 4% / 75%	Mixed strides
loop_site_145	grid_intersect	grid.cpp:562	No information available	21% / 4% / 75%	Mixed strides
loop_site_135	initialize_2D_buffer	find_hotspots.cpp:92	No information available	42% / 0% / 58%	Mixed strides

Memory Access Patterns Report | Correctness Report

ID	Stride	Type	Source	Modules	Alignment
⊞P17	8	Constant stride	intersect.cpp:141	find_hotspots.exe	
⊟P19	8	Constant stride	intersect.cpp:141	find_hotspots.exe	
⊞P1.	0	Unit stride		d_hotspots.exe	
⊞P1.	0	Unit stride	intersect.cpp:141	find_hotspots.exe	
⊞P2.	-8; -2; 0; 1; 2; ...	Variable stride			
⊞P2.	-8; -2; 0; 1; 2; ...	Variable stride			
⊞P21	4	Constant stride	intersect.cpp:142	find_hotspots.exe	

```
139
140       intstruct->num++;
141       intstruct->list[intstruct->num].obj = obj;
142       intstruct->list[intstruct->num].t = t;
143     }
```

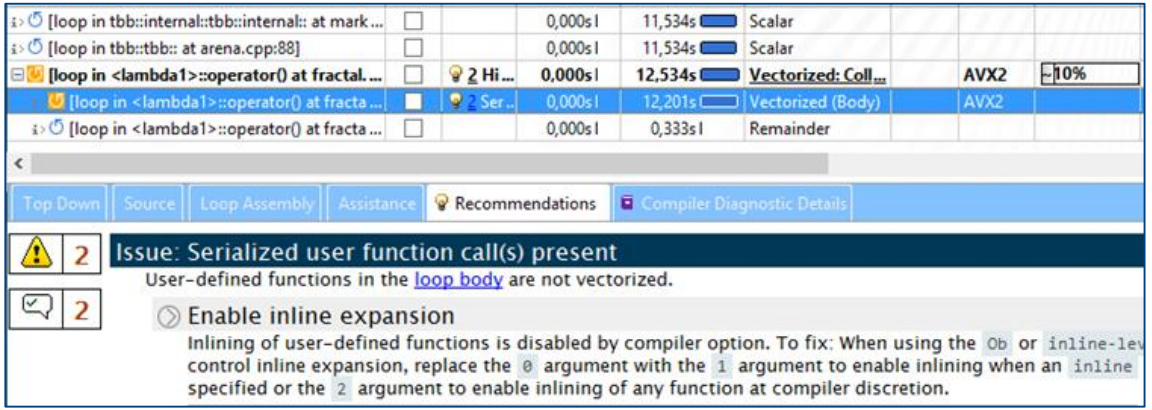
Constant stride, “Array of Structures”

Unit stride access

Variable or random access

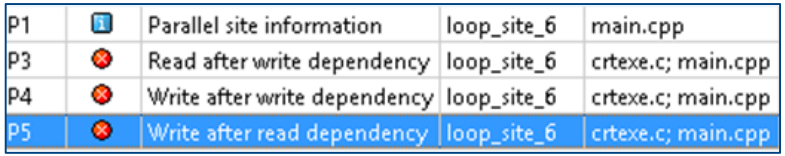
Intel Advisor XE

Все диагностики в одном месте

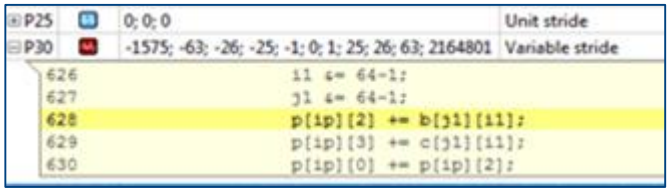


Рекомендации по оптимизации

Проверка зависимостей

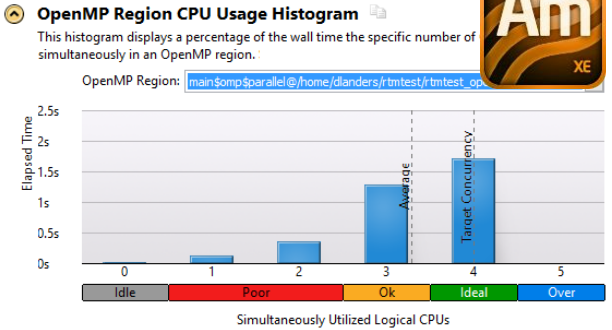


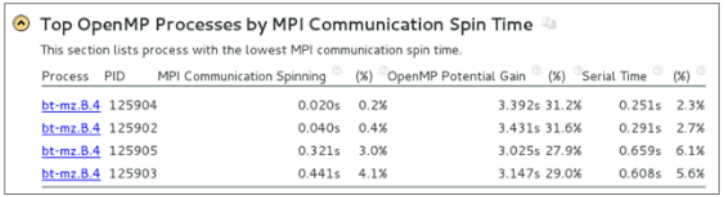
Анализ шаблонов доступа

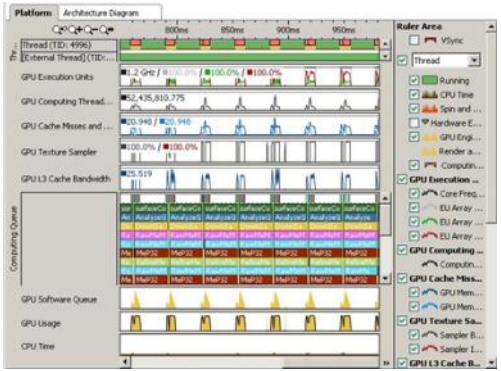


Intel® VTune™ Amplifier XE

- Масштабируемость OpenMP
- Гибридный анализ с MPI
- OpenCL & GPU анализ
- Улучшенный анализ доступа к памяти
- Быстрее и проще
- Добавлена поддержка VM
- Последние процессоры & ОС







Intel® VTune™ Amplifier XE

Эффективное использование OpenMP

Типичные вопросы:

Q: *“Я добавил прагмы в код, но почему ускорение далеко не линейно?”*

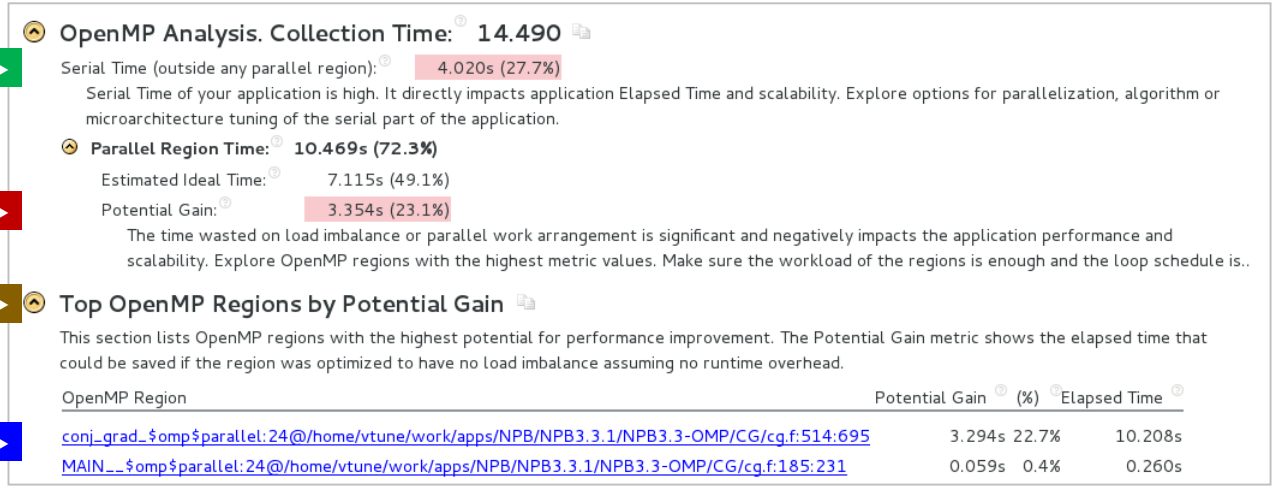
A: Неэффективное распараллеливание

Q: *“Я запускаю приложение на системе с большим числом ядер, но почему оно выполняется медленнее?”*

A: Проблема масштабируемости

Intel® VTune™ Amplifier XE

Эффективное использование OpenMP



OpenMP Analysis. Collection Time: 14.490

1) ▶ Serial Time (outside any parallel region): 4.020s (27.7%)

Serial Time of your application is high. It directly impacts application Elapsed Time and scalability. Explore options for parallelization, algorithm or microarchitecture tuning of the serial part of the application.

Parallel Region Time: 10.469s (72.3%)

Estimated Ideal Time: 7.115s (49.1%)

2) ▶ Potential Gain: 3.354s (23.1%)

The time wasted on load imbalance or parallel work arrangement is significant and negatively impacts the application performance and scalability. Explore OpenMP regions with the highest metric values. Make sure the workload of the regions is enough and the loop schedule is..

3) ▶ Top OpenMP Regions by Potential Gain

This section lists OpenMP regions with the highest potential for performance improvement. The Potential Gain metric shows the elapsed time that could be saved if the region was optimized to have no load imbalance assuming no runtime overhead.

OpenMP Region	Potential Gain	(%)	Elapsed Time
4) ▶ conj_grad_ompparallel:24@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:514:695	3.294s	22.7%	10.208s
MAIN__ompparallel:24@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:185:231	0.059s	0.4%	0.260s

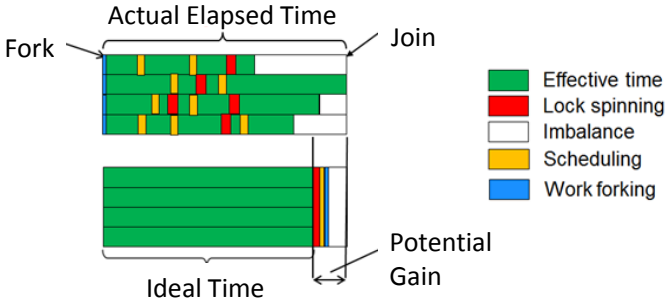
Ответы на интересующие вопросы:

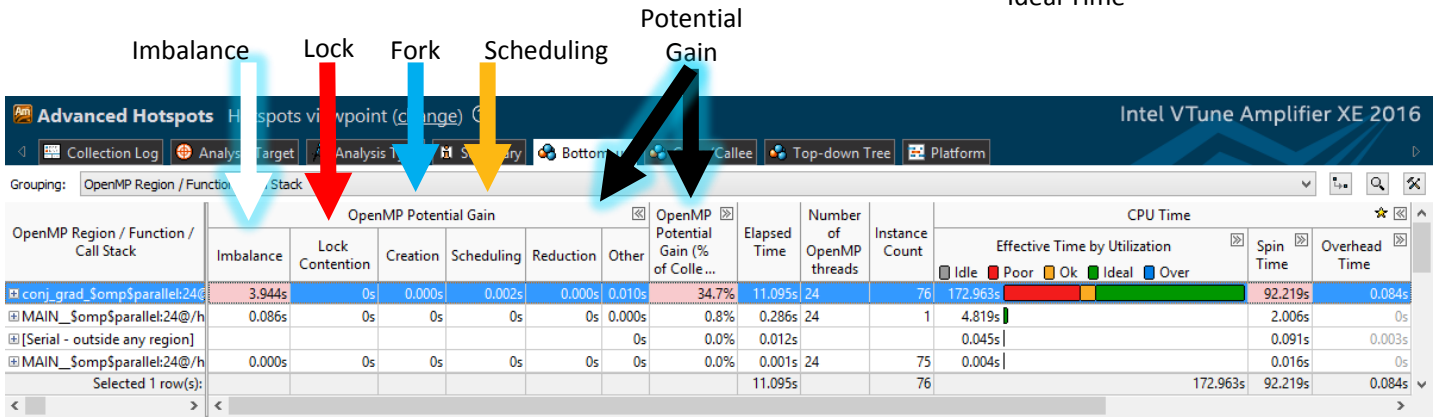
1) Насколько велика последовательная часть приложения?
2) Сколько можно «выжать» при оптимизации OpenMP?
3) На какие OpenMP регионы/циклы/барьеры обратить внимание?
4) Что неэффективного в каждом регионе?

Intel® VTune™ Amplifier XE

Эффективное использование OpenMP

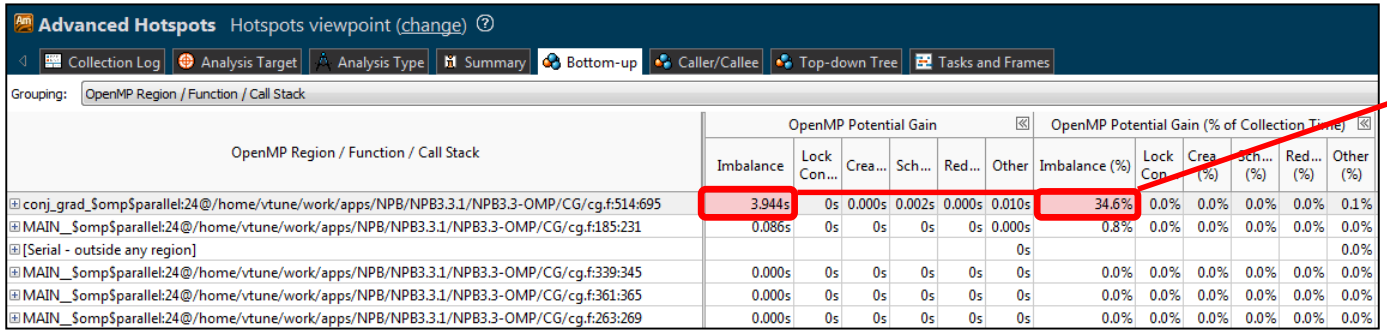
- Показывает то, что важно
  - Какой регион неэффективен?
  - Насколько можем улучшить?
  - Почему неэффективен?
    Дисбаланс? Планировка? Lock spinning?
  - Поддержка Intel® Xeon Phi





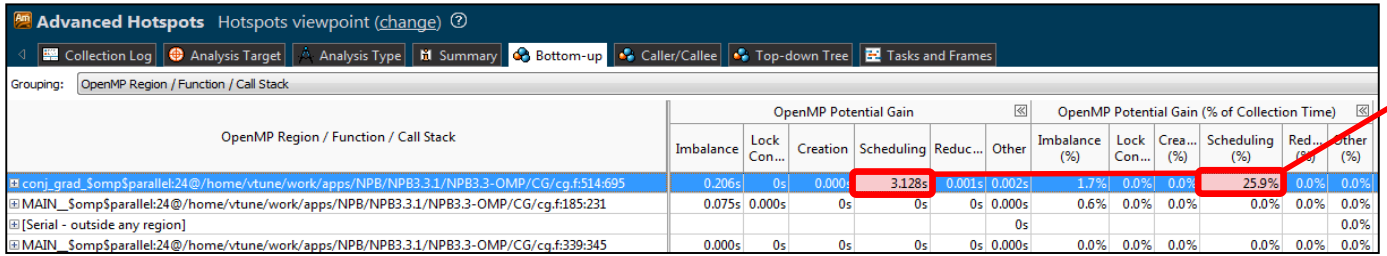
Intel® VTune™ Amplifier XE

Определяем неэффективные регионы OpenMP



OpenMP Region / Function / Call Stack	OpenMP Potential Gain						OpenMP Potential Gain (% of Collection Time)					
	Imbalance	Lock Con...	Crea...	Sch...	Red...	Other	Imbalance (%)	Lock Con...	Crea... (%)	Sch... (%)	Red... (%)	Other (%)
conj_grad_ompparallel:24@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:514:695	3.944s	0s	0.000s	0.002s	0.000s	0.010s	34.6%	0.0%	0.0%	0.0%	0.0%	0.1%
MAIN__ompparallel:24@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:185:231	0.086s	0s	0s	0s	0s	0.000s	0.8%	0.0%	0.0%	0.0%	0.0%	0.0%
[Serial - outside any region]						0s						0.0%
MAIN__ompparallel:24@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:339:345	0.000s	0s	0s	0s	0s	0s	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
MAIN__ompparallel:24@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:361:365	0.000s	0s	0s	0s	0s	0s	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
MAIN__ompparallel:24@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:263:269	0.000s	0s	0s	0s	0s	0s	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%

Дисбаланс



OpenMP Region / Function / Call Stack	OpenMP Potential Gain						OpenMP Potential Gain (% of Collection Time)					
	Imbalance	Lock Con...	Creation	Scheduling	Reduc...	Other	Imbalance (%)	Lock Con...	Crea... (%)	Scheduling (%)	Red... (%)	Other (%)
conj_grad_ompparallel:24@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:514:695	0.206s	0s	0.000s	3.128s	0.001s	0.002s	1.7%	0.0%	0.0%	25.9%	0.0%	0.0%
MAIN__ompparallel:24@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:185:231	0.075s	0.000s	0s	0s	0s	0.000s	0.6%	0.0%	0.0%	0.0%	0.0%	0.0%
[Serial - outside any region]						0s						0.0%
MAIN__ompparallel:24@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:339:345	0.000s	0s	0s	0s	0s	0.000s	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%

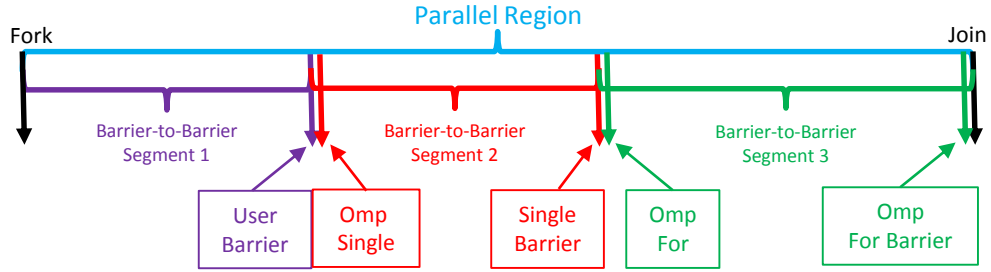
Возможная причина: Накладные расходы от динамической планировки

Intel® VTune™ Amplifier XE

Определяем причину проблем

- Подробный анализ (Barrier to Barrier)
  - Рассматриваем каждый сегмент отдельно
  - Проще видеть возможности к улучшению

```
#pragma omp parallel
{
   …
#pragma omp barrier
#pragma omp single
      {
         …
      {
#pragma omp for
      {
```



Grouping: OpenMP Region / OpenMP Barrier-to-Barrier Segment / Function / Call Stack

OpenMP Region / OpenMP Barrier-to-Barrier Segment / Function / Call Stack	OpenMP Potential Gain						OpenMP Potential Gain (% of Collection ...
	Imbalance	Lock Contention	Creation	Scheduling	Reduction	Other	
3.3-OMP/CG/cg.f:514:695	3.944s	0s	0.000s	0.002s	0.000s	0.010s	34.7%
\PB3.3.1/NPB3.3-OMP/CG/cg.f:580	3.725s	0s	0s	0.000s	0s	0.008s	32.8%
\PB3.3.1/NPB3.3-OMP/CG/cg.f:683	0.149s	0s	0s	0s	0s	0.000s	1.3%
\PB3.3.1/NPB3.3-OMP/CG/cg.f:664	0.014s	0s	0s	0.000s	0s	0.000s	0.1%

Intel® VTune™ Amplifier XE

Он стал быстрее!

	Initial Release **2015 Mid-Size Data**	Initial Release **2015 Large Data**	Initial Release **2016 Both Sizes**
Open Summary	10-40 seconds	1-2 minutes	~4-5 seconds
Open Timeline	10-40 seconds	2-5 minutes	~1-3 seconds
Zoom Timeline	5-10 seconds	1-2 minutes	~ 1 second
Grid Node Expand	5-15 seconds	>1 minute	<1 – 2 seconds
Finalization			Usually <2x slower Sometimes faster

- При работе с большими размерами данных получаем наибольшее ускорение.
- При маленьких размерах – незначительное улучшение.

Совет: для лучшей производительности избегайте медленной графики VNC. Запускайте UI локально. Импортируйте данные с удаленной системы.

Данные получены по результатам внутреннего тестирования. Ваши результаты могут отличаться.
Результаты зависят от характеристик системы, приложения и размера дынных.

Отзывы разработчиков

"Intel® VTune™ Amplifier XE analyzes complex code and helps us identify bottlenecks rapidly. By using it and other Intel® Software Development Tools, we were able to improve PIPESIM performance up to 10 times compared with the previous software version."

Rodney Lessard
Senior Scientist
Schlumberger

"**Intel® Advisor XE** has been extremely helpful in identifying the best pieces of code for parallelization. We can save several days of manual work by targeting the right loops. At the same time, we can use Advisor to find potential thread safety issues to help avoid problems later on."

Carlos Boneti
HPC software engineer,
Schlumberger

Intel® Inspector XE has dramatically sped up our ability to find/fix memory problems and track down difficult to isolate threading errors before our packages are released to the field.

Peter von Kaenel, Director, Software Development, ***Harmonic Inc.***

Call to Action

- Пробуем сегодня!

  https://software.intel.com/en-us/intel-parallel-studio-xe/try-buy

- Intel® Premier Support

- Ваш отзыв по использованию

Intel® Parallel Studio XE 2016 важен!

Q&A

Legal Disclaimer & Optimization Notice

INFORMATION IN THIS DOCUMENT IS PROVIDED “AS IS”. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO THIS INFORMATION INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

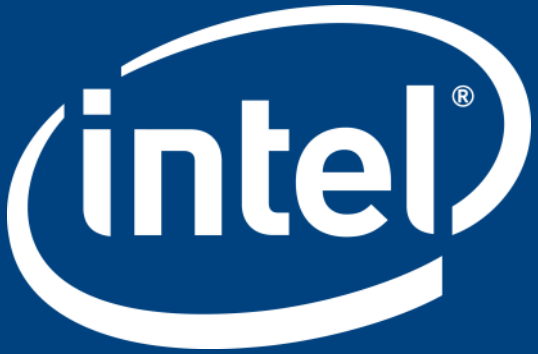
Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products.

Copyright © 2015, Intel Corporation. All rights reserved. Intel, Pentium, Xeon, Xeon Phi, Core, VTune, Cilk, and the Intel logo are trademarks of Intel Corporation in the U.S. and other countries.

Optimization Notice

Intel’s compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

Notice revision #20110804


intel®