

Hands-on: CloverLeaf_OpenACC w/ Score-P (II)

Markus Geimer
Jülich Supercomputing Centre



Congratulations!?

- If you made it this far, you successfully used Score-P to
 - instrument the application
 - analyze its execution with a summary measurement, and
 - examine it with one of the interactive analysis report explorer GUIs
- ... revealing the call-path profile annotated with
 - the “Time” metric
 - Visit counts
 - MPI message statistics (bytes sent/received)
- ... but how **good** was the measurement?
 - The measured execution produced the desired valid result
 - and the execution took slightly longer than expected.
 - even when ignoring measurement start-up/completion, therefore
 - it was probably dilated by instrumentation/measurement overhead

CloverLeaf_OpenACC summary analysis result scoring

```
% scorep-score scorep_clover_leaf_8_sum/profile.cubex
```

Estimated aggregate size of event trace:

Estimated requirements for largest trace buffer (max_buf):

Estimated memory requirements (SCOREP_TOTAL_MEMORY):

(hint: When tracing set SCOREP_TOTAL_MEMORY=15MB to avoid intermediate flushes or reduce requirements using USR regions filters.)

flt	type	max_buf[B]	visits	time[s]	time[%]	time/visit[us]	region
	ALL	10,626,843	1,926,637	455.01	100.0	236.17	ALL
	CUDA	6,396,588	837,005	396.15	87.1	473.30	CUDA
	OPENACC	3,315,936	899,628	0.74	0.2	0.83	OPENACC
	COM	565,734	151,708	0.22	0.0	1.43	COM
	MPI	328,420	31,904	19.99	4.4	626.72	MPI
	USR	25,116	6,384	37.89	8.3	5935.49	USR
	SCOREP	41	8	0.01	0.0	651.80	SCOREP

74MB

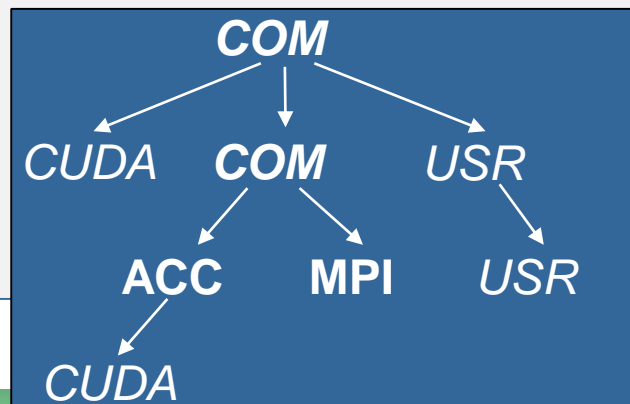
11MB

15MB

- Report scoring as textual output

74 MB total memory
15 MB max per MPI process

- Region/callpath classification
 - MPI** pure MPI functions
 - OMP** pure OpenMP regions
 - OPENACC** pure OpenACC regions
 - CUDA** API regions & kernels
 - USR** user source routines + API
 - COM** "combined" USR+MPI/ACC/etc
 - SCOREP** measurement library
 - ALL** aggregate of all region types



CloverLeaf_OpenACC summary analysis detailed report

```
% scorep-score -r scorep_clover_leaf_8_sum/profile.cubex
```

flt	type	max_buf[B]	visits	time[s]	time[%]	time/visit[us]	region
	ALL	7,316,653	1,027,141	459.68	100.0	447.53	ALL
	CUDA	6,402,334	837,137	398.97	86.8	476.59	CUDA
	COM	565,734	151,708	0.71	0.2	4.69	COM
	MPI	328,420	31,904	22.09	4.8	692.49	MPI
	USR	25,116	6,384	37.90	8.2	5936.08	USR
	SCOREP	41	8	0.01	0.0	957.52	SCOREP
	CUDA	1,088,100	294,580	103.15	22.7	350.16	cuStreamSynchronize
	CUDA	819,984	128,620	0.70	0.2	5.46	cuLaunchKernel
	CUDA	444,184	128,628	127.03	27.9	987.54	COMPUTE_IDLE
	CUDA	406,222	8,044	0.03	0.0	3.66	pack_kernel_module_clover_pack_message_right_170_gpu
	CUDA	406,222	8,044	0.02	0.0	2.68	pack_kernel_module_clover_unpack_message_right_222_gpu
	CUDA	406,222	12,066	0.04	0.0	2.94	pack_kernel_module_clover_pack_message_top_273_gpu
	CUDA	406,222	12,066	0.03	0.0	2.63	pack_kernel_module_clover_unpack_message_top_325_gpu
	CUDA	406,222	8,044	0.03	0.0	3.65	pack_kernel_module_clover_pack_message_left_67_gpu
	CUDA	406,222	8,044	0.02	0.0	2.64	pack_kernel_module_clover_unpack_message_left_120_gpu
	CUDA	406,222	12,066	0.04	0.0	2.96	pack_kernel_module_clover_pack_message_bottom_377_gpu
	CUDA	406,222	12,066	0.03	0.0	2.65	pack_kernel_module_clover_unpack_message_bottom_430_gpu
	CUDA	253,396	67,981	0.05	0.0	0.68	cuPointerGetAttributes
	CUDA	163,748	42,340	33.82	7.4	798.75	cuMemcpyHtoDAsync_v2
	CUDA	161,720	41,716	24.17	5.3	579.28	cuMemcpyDtoHAsync_v2
	MPI	139,641	10,460	0.02	0.0	1.63	MPI_Irecv
	MPI	139,641	10,460	0.03	0.0	2.77	MPI_Isend
[...]							
	OPENACC	52,286	8,044	0.01	0.0	0.83	acc_data_enter@pack_kernel.f90:164
	OPENACC	52,286	8,044	0.00	0.0	0.59	acc_wait@pack_kernel.f90:164
[...]							

region names
(with full signatures)

Score-P filtering: Automatic generation of filter files



- **Basic usage:** `scorep-score -g`
default heuristic targets:
 - Buffer usage: relevancy
 - Time per visits: overhead
- Creates annotated filter file:
 - `initial_scorep.filter`
 - Repeated calls create backups
 - Usage with `-f <file>` results in inclusion
- **Objective:**
 - Starting point for filtering
 - Syntax introduction

```
-g [<list>]
```

Generation of an initial filter file with the name 'initial_scorep.filter'. A valid parameter list has the form `KEY=VALUE[,KEY=VALUE]*`. By **default**, uses the following control parameters:

```
`bufferpercent=1,timepervisit=1`
```

A region is included in the filter file (i.e., excluded from measurement) if it matches all of the given conditions, with the following keys:

- ``bufferpercent`` : estimated memory requirements exceed the given threshold in percent of the total estimated trace buffer requirements
- ``bufferabsolute`` : estimated memory requirements exceed the given absolute threshold in MB
- ``visits`` : number of visits exceeds the given threshold
- ``timepervisit`` : time per visit value is below the given threshold in microseconds
- ``type`` : region type matches the given value (allowed: 'usr', 'com', 'both')

Score-P filtering: Automatic generation of filter files



```
. . .
#
# Generated with the following parameters:
# - A region has to use at least 1% of the estimated trace buffer.
# - A region has to have a time/visits value of less than 1 us.
# - A region that is of type USR.
#
# The file contains comments for each region providing additional information
# regarding the respective region.
# The common prefix for the files is:
# '/'
#
# Please refer to the Score-P user guide for more options on filtering.
SCOREP_REGION_NAMES_BEGIN
EXCLUDE
  # type=USR max_buf= 63,576,240 visits=16,628,248, time= 2.17s ( 0.4%), time/visit= 0.13us
  # name='bool std::operator< <char, std::char_traits<char>, std::allocator<char> >(std::__cxx11:: [...]
  # file='p/software/juwelsbooster/stages/2022/software/GCCcore/11.2.0/include/c++/11.2.0/bits/ [...]
  MANGLED _ZStltIcSt11char_traitsIcESaIcEEbRKNSt7__cxx1112basic_stringIT_T0_T1_EESA_
. . .
SCOREP_REGION_NAMES_END
```

Introductory preamble

Used generation parameters
(either default or explicit)

Common path prefix to simplify
file paths in entries

Annotated entry with comments
containing scorep-score stats

Additional entries

CloverLeaf_OpenACC summary analysis filter preview

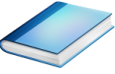
```
% scorep-score -r -f scorep.filter scorep_clover_leaf_8_sum/profile.cubex
```

flt	type	max_buf[B]	visits	time[s]	time[%]	time/visit[us]	region
*	ALL	8,949,895	1,476,948	287.21	63.1	194.46	ALL-FLT
*	CUDA	4,719,640	387,316	228.36	50.2	589.59	CUDA-FLT
-	OPENACC	3,315,936	899,628	0.74	0.2	0.83	OPENACC-FLT
+	FLT	1,676,948	449,689	167.79	36.9	373.13	FLT
*	COM	565,734	151,708	0.22	0.0	1.43	COM-FLT
-	MPI	328,420	31,904	19.99	4.4	626.72	MPI-FLT
*	USR	25,116	6,384	37.89	8.3	5935.49	USR-FLT
-	SCOREP	41	8	0.01	0.0	651.80	SCOREP-FLT
+	CUDA	1,088,100	294,580	103.15	22.7	350.16	cuStreamSynchronize
-	CUDA	819,984	128,620	0.70	0.2	5.46	cuLaunchKernel
-	CUDA	444,184	128,628	127.03	27.9	987.54	COMPUTE IDLE
-	CUDA	406,222	8,044	0.03	0.0	3.66	pack_kernel_module_clover_pack_message_right_170_gpu
-	CUDA	406,222	8,044	0.02	0.0	2.68	pack_kernel_module_clover_unpack_message_right_222_gpu
-	CUDA	406,222	12,066	0.04	0.0	2.94	pack_kernel_module_clover_pack_message_top_273_gpu
-	CUDA	406,222	12,066	0.03	0.0	2.63	pack_kernel_module_clover_unpack_message_top_325_gpu
-	CUDA	406,222	8,044	0.03	0.0	3.65	pack_kernel_module_clover_pack_message_left_67_gpu
-	CUDA	406,222	8,044	0.02	0.0	2.64	pack_kernel_module_clover_unpack_message_left_120_gpu
-	CUDA	406,222	12,066	0.04	0.0	2.96	pack_kernel_module_clover_pack_message_bottom_3
-	CUDA	406,222	12,066	0.03	0.0	2.65	pack_kernel_module_clover_unpack_message_bottom
+	CUDA	253,396	67,981	0.05	0.0	0.68	cuPointerGetAttributes
+	CUDA	163,748	42,340	33.82	7.4	798.75	cuMemcpyHtoDAsync_v2
+	CUDA	161,720	41,716	24.17	5.3	579.28	cuMemcpyDtoHAsync_v2
-	MPI	139,641	10,460	0.02	0.0	1.63	MPI_Irecv
-	MPI	139,641	10,460	0.03	0.0	2.77	MPI_Isend

max_buf reductions estimated with filter

region names matching filter rules marked with '+'

Score-P filtering



```
% cat ../config/scorep.filter
SCOREP_REGION_NAMES_BEGIN
EXCLUDE
  binvrhs*
  matmul_sub*
  matvec_sub*
  exact_solution*
  binvrhs*
  lhs*init*
  timer_*
SCOREP_REGION_NAMES_END

% export SCOREP_FILTERING_FILE=\
../config/scorep.filter
```

Region name
filter block
using wildcards

Apply filter

- Filtering by source file name
 - All regions in files that are excluded by the filter are ignored
- Filtering by region name
 - All regions that are excluded by the filter are ignored
 - Overruled by source file filter for excluded files
- Apply filter by
 - exporting `SCOREP_FILTERING_FILE` environment variable
- Apply filter at
 - Run-time
 - Compile-time (GCC-plugin, Intel compiler)
 - Add cmd-line option `--instrument-filter`
 - No overhead for filtered regions but recompilation

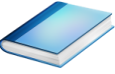
Source file name filter block



- Keywords
 - Case-sensitive
 - SCOREP_FILE_NAMES_BEGIN, SCOREP_FILE_NAMES_END
 - Define the source file name filter block
 - Block contains EXCLUDE, INCLUDE rules
 - EXCLUDE, INCLUDE rules
 - Followed by one or multiple white-space separated source file names
 - Names can contain bash-like wildcards *, ?, []
 - Unlike bash, * may match a string that contains slashes
- EXCLUDE, INCLUDE rules are applied in sequential order
- Regions in source files that are excluded after all rules are evaluated, get filtered

```
# This is a comment
SCOREP_FILE_NAMES_BEGIN
# by default, everything is included
EXCLUDE */foo/bar*
INCLUDE */filter_test.c
SCOREP_FILE_NAMES_END
```

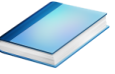
Region name filter block



- Keywords
 - Case-sensitive
 - SCOREP_REGION_NAMES_BEGIN,
SCOREP_REGION_NAMES_END
 - Define the region name filter block
 - Block contains EXCLUDE, INCLUDE rules
 - EXCLUDE, INCLUDE rules
 - Followed by one or multiple white-space separated region names
 - Names can contain bash-like wildcards *, ?, []
- EXCLUDE, INCLUDE rules are applied in sequential order
- Regions that are excluded after all rules are evaluated, get filtered

```
# This is a comment
SCOREP_REGION_NAMES_BEGIN
# by default, everything is included
EXCLUDE *
INCLUDE bar foo
        baz
        main
SCOREP_REGION_NAMES_END
```

Region name filter block, mangling



- Name mangling
 - Filtering based on names seen by the measurement system
 - Dependent on compiler
 - Actual name may be mangled
- `scorep-score` names as starting point
(e.g. `matvec_sub_`)
 - Use `*` for Fortran trailing underscore(s) for portability
 - Use `?` and `*` as needed for full signatures or overloading

```
void bar(int* a) {  
    *a++;  
}  
int main() {  
    int i = 42;  
    bar(&i);  
    return 0;  
}
```

```
# filter bar:  
# for gcc-plugin, scorep-score  
# displays 'void bar(int*)',  
# other compilers may differ  
  
SCOREP_REGION_NAMES_BEGIN  
    EXCLUDE void?bar(int?)  
SCOREP_REGION_NAMES_END
```

Score-P user instrumentation API



- No replacement for automatic compiler instrumentation
- Can be used to further subdivide functions
 - E.g., multiple loops inside a function
- Can be used to partition application into coarse grain phases
 - E.g., initialization, solver, & finalization
- Enabled with `--user` flag to Score-P instrumenter
- Available for Fortran / C / C++

Score-P user instrumentation API (Fortran)



```
#include "scorep/SCOREP_User.inc"

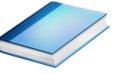
subroutine foo(...)
  ! Declarations
  SCOREP_USER_REGION_DEFINE( solve )

  ! Some code...
  SCOREP_USER_REGION_BEGIN( solve, "<solver>", \
                           SCOREP_USER_REGION_TYPE_LOOP )

  do i=1,100
    [...]
  end do
  SCOREP_USER_REGION_END( solve )
  ! Some more code...
end subroutine
```

- Requires processing by the C preprocessor
 - For most compilers, this can be automatically achieved by having an uppercase file extension, e.g., `main.F` or `main.F90`

Score-P user instrumentation API (C/C++)

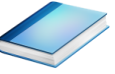


```
#include "scorep/SCOREP_User.h"

void foo()
{
    /* Declarations */
    SCOREP_USER_REGION_DEFINE( solve )

    /* Some code... */
    SCOREP_USER_REGION_BEGIN( solve, "<solver>",
                             SCOREP_USER_REGION_TYPE_LOOP )
    for (i = 0; i < 100; i++)
    {
        [...]
    }
    SCOREP_USER_REGION_END( solve )
    /* Some more code... */
}
```

Score-P user instrumentation API (C++)



```
#include "scorep/SCOREP_User.h"

void foo()
{
    // Declarations

    // Some code...
    {
        SCOREP_USER_REGION( "<solver>",
                           SCOREP_USER_REGION_TYPE_LOOP )
        for (i = 0; i < 100; i++)
        {
            [...]
        }
    }
    // Some more code...
}
```


Score-P measurement control API



- Can be used to temporarily disable measurement for certain intervals
 - Annotation macros ignored by default
 - Enabled with `--user` flag

```
#include "scorep/SCOREP_User.inc"

subroutine foo(...)
  ! Some code...
  SCOREP_RECORDING_OFF()
  ! Loop will not be measured
  do i=1,100
    [...]
  end do
  SCOREP_RECORDING_ON()
  ! Some more code...
end subroutine
```

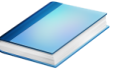
Fortran (requires C preprocessor)

```
#include "scorep/SCOREP_User.h"

void foo(...) {
  /* Some code... */
  SCOREP_RECORDING_OFF()
  /* Loop will not be measured */
  for (i = 0; i < 100; i++) {
    [...]
  }
  SCOREP_RECORDING_ON()
  /* Some more code... */
}
```

C / C++

Enriching measurements with performance counters



- Record metrics from PAPI:

```
% export SCOREP_METRIC_PAPI=PAPI_TOT_CYC
% export SCOREP_METRIC_PAPI_PER_PROCESS=PAPI_L3_TCM
```

- Use PAPI tools to get available metrics and valid combinations:

```
% papi_avail
% papi_native_avail
```

- Record metrics from Linux perf:

```
% export SCOREP_METRIC_PERF=cpu-cycles
% export SCOREP_METRIC_PERF_PER_PROCESS=LLC-load-misses
```

- Use the `perf` tool to get available metrics and valid combinations:

```
% perf list
```

- Write your own metric plugin

- Repository of available plugins: <https://github.com/score-p>

Only the master thread records the metric (assuming all threads of the process access the same L3 cache)

Further information

- Community instrumentation & measurement infrastructure
 - Instrumentation (various methods)
 - Basic and advanced profile generation
 - Event trace recording
 - Online access to profiling data
- Available under 3-clause BSD open-source license
- Documentation & Sources:
 - <https://www.score-p.org>
- User guide also part of installation:
 - `<prefix>/share/doc/scorep/{pdf,html}/`
- Support and feedback: support@score-p.org
- Subscribe to news@score-p.org, to be kept informed