Introduction to Intel Advisor at NERSC

Mathieu Lobet, Tuomas Koskela

NERSC
November 3 2016
Purpose of Intel Advisor

Intel Advisor is composed of Vectorization advisor and Threading advisor that help to ensure that your code realizes full performance potential.

Vectorization advisor is a vectorization optimization tool:

- Help to identify time-consuming loops that can benefit from vectorization or already vectorized
- Help to identify vectorization and efficiency issues (dependencies, spilling, memory access...) and propose solutions
- Help to ensure that vectorization is safe and quantify effects of vectorization (vectorization efficiency, roofline performance model)
Factors that prevent vectorization

Loop-carried dependencies/data dependencies

```c
For(i = 0 ; i < N ; i++ ) {
}
```

Call to sequential functions

```c
For(i = 0 ; i < N ; i++ ) {
    sequential_function(i);
}
```

Pointer aliasing

```c
Void function(float * A, float * B) {
    for(i = 0 ; i < N ; i++ ) {
        A[i] = B[i] * C;
    }
}
```

Inner/Outer loops

```c
Do i=1,N
    A(i) = B(i) + C(i)
    S(1:M) = A(1:M)
Enddo
```
Factors that slow down vectorization

Indirect/strided memory access

```
For(i = 0 ; i < N ; i++ ) {
    A[C[i]] = B[D[i]];
}
```

Memory latency/throughput (low flop/byte ratio)

```
For(i = 0 ; i < very big ; i++ ) {
    A[i] = c*B[i];
}
```

Small trip count not multiple of vector length

```
For(i = 0 ; i < small; i++ ) {
    A[i] = c*B[i];
}
```

Branching, divisions, exp...

```
For(i = 0 ; i < small; i++ ) {
    if (B[i] > 0) {
        A[i] = C[i]/B[i];
    }
}
```
Very Good references for Advisor

Intel Advisor Linux presentation (description of the different surveys, links toward specific command documentation):

Intel Advisor Getting Started/FAQ:

The NERSC website:
How to perform a collection

Compile for advisor:

Cray:  
CC -dynamic -g -openmp -O3 -xMIC-AVX512 mycode.c -o $APPNAME

Intel:  
CC -g -openmp -O3 -xMIC-AVX512 mycode.c -o $APPNAME

Perform analysis by command line:

salloc -N 1 -p debug -t 01:00:00
module load advisor

srn -n 1 -c 1 advixe-cl - collect $ANALYSIS
--project-dir $ANALYSIS_DIR
--search-dir src:r=$SRCDIR
--trace-mpi ./$APPNAME

Use the Gui interface for performing analysis on a login node:

salloc -N 1 -p debug -t 01:00:00
module load advisor

advixe-gui ./$ANALYSIS_DIR
Different kind of collections

srun -n 1 -c 1 advixe-cl -collect $ANALYSIS --project-dir $ANALYSIS_DIR --search-dir src=r=$SRCDIR --trace-mpi ./$APPNAME

$ANALYSIS=survey: general overview of the performances and the vectorization state of the code.

$ANALYSIS=tripcounts: improves the survey by dynamically exploring loop iteration execution and propose better decisions about your vectorization strategy. It measures #FLOP count and cumulative data traffic necessary for the Roofline performance model.

$ANALYSIS=dependencies: refine analysis by checking for real data dependencies in loops the compiler did not vectorize because of assumed dependencies.

$ANALYSIS=map: (Memory Access Pattern) refine analysis by checking for various memory issues, such as non-contiguous memory accesses and unit stride vs. non-unit stride accesses.

Other options

-no-auto-finalize: Results are not finalize after the run. This is useful for collections on KNL that takes longer time than on Haswell. Finalization can then be performed on Haswell or when opening the results via the GUI.

-flops-and-masks: take into account the masks in the vector operations on KNL.
Analyzing the results

Analyze the results via the GUI:

Module load advisor
advixe-gui ./$ANALYSIS_DIR
The cache-aware roofline model [2]: Roofline automation in Intel Advisor:

“Roofline is a visually intuitive performance model used to bound the performance of various numerical methods and operations running on multicore, manycore, or accelerator processor architectures.” [1]

- Requires the survey and the tripcounts analysis
- Computation of the roofline: u-bench-based
- AVX-512 mask-aware
- Break-down by loops or functions
- Measure L1 <-> Register traffic: what CPU demands from memory sub-system to make a computation
- Cumulative traffic through L1/L2/LLC/DRAM/MCDRAM
- GUI for quick viewing

Very Good references for Advisor

Let’s have a short demonstration
Thank You