Intel Advisor on Cori

Charlene Yang

Application Performance Group
cjyang@lbl.gov
Old approach -- pen and paper

GPP kernel on KNL

- AI (DRAM): 66.39
- AI (MCDRAM): 2.70
- AI (L2): 1.78
- AI (L1): 1.01
- Performance: 171.960 GFLOPS/s

Arithmetic Intensity

Attainable Flop/s

171.96GFLOP/s

2.7TFLOP/s

1.01 1.78 2.70 66.39

L1 12.2TB/s
L2 2.0TB/s
MCDRAM 368.5GB/s
DDR 77.0GB/s
Intel® Advisor: automatic and refined
Intel® Advisor: code analytics

Trip Counts
Intel® Advisor calculates approximate value of the source (scalar) loop trip counts according to the following formula:

\[ \text{TC崖底} = V_{\text{崖底}} \times T_{\text{崖底}} \]

Where:
- \( V_{\text{崖底}} \) Vector length of loop body, equals to 4.
- \( T_{\text{崖底}} \) Trip count of vectorized body part, equals to 5.

Call Count: 330187500
Iteration Duration: < 0.001s

Statistics for FLOPS And Data Transfers

- Self GFLOPS: 15.28575
- Total GFLOPS: 15.28575
- Self AI: 0.46429
- Total AI: 0.46429
- Self GFLOP: 86.18875
- Total GFLOP: 86.18875

Self Elapsed Time: 5.769s
Total Elapsed Time: 5.769s

Data transfers between CPU and memory sub-system (total traffic, including L1, L2, LLC and DRAM traffic):
- In Giga Bytes Of Functions Called In The Loop Or Function: 189,94500
- In Giga Bytes Of Functions Called In The Loop Or Function: 189,94500
- In Giga Bytes Per Second: 32,92315
- In Bytes Per Loop Iteration: 112
## Intel® Advisor: 2-step data collection

<table>
<thead>
<tr>
<th>Step 1: Survey (-collect survey)</th>
<th>Overhead</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Provide <strong>#Seconds</strong></td>
<td>1x</td>
</tr>
<tr>
<td>- Root access not needed</td>
<td></td>
</tr>
<tr>
<td>- User mode sampling, non-intrusive.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 2: FLOPS (-collect tripcounts –flops)</th>
<th>Overhead</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Provide <strong>#FLOP</strong>, <strong>#Bytes</strong>, AVX-512 Mask</td>
<td>3-5x</td>
</tr>
<tr>
<td>- Root access not needed</td>
<td></td>
</tr>
<tr>
<td>- Precise, instrumentation based, count number of instructions</td>
<td></td>
</tr>
</tbody>
</table>

### Roofline:

<table>
<thead>
<tr>
<th>Axis X: <strong>AI</strong> = <strong>#FLOP / #Bytes</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Axis Y: <strong>FLOP/S</strong> = <strong>#FLOP</strong> (mask aware) / <strong>#Seconds</strong></td>
<td></td>
</tr>
</tbody>
</table>
Setup on Cori

• Compile with -g to get debugging info
  
  cc -g -dynamic -openmp -O2 -o mycode.exe mycode.c

• Cache-Aware Roofline Model (CARM)
  
  module load advisor/2018.up1

• Integrated Roofline Model (Cache Simulator)
  
  module load advisor/2018.integrated_roofline.up1

• Incompatible GUI for regular and integrated Advisor
Run Advisor on Cori

- Start an interactive session on a KNL node
  
  ```bash
  salloc --qos=interactive -C knl -N 1 -t hh:mm:ss -A <your_account>
  ```

- To collect data for roofline, do two collections: **survey** and **tripcounts**.
  
  ```bash
  srun -n <num-of-ranks> -c <num_of_cores_per_rank> advixe-cl -v
  -collect survey -no-auto-finalize -project-dir=<same_dir_name>
  -data-limit=0 -- <your_executable>
  ```

  ```bash
  srun -n <num-of-ranks> -c <num_of_cores_per_rank> advixe-cl -v
  -collect tripcounts -flops -no-auto-finalize -project-dir=<same_dir_name>
  -data-limit=0 -- <your_executable>
  ```

- Run on the Lustre filesystem `$SCRATCH`
- Finalization is expensive especially on KNL: do it offline!
Pack/View results

- Pack results/source file/binary (already packed in demo)

```bash
advixe-cl --snapshot --project-dir <same_dir_name> --pack --cache-sources --cache-binaries -- <target_file_name>
```

- Load module

```bash
module load advisor/2018.integrated_roofline.up1
module show advisor/2018.integrated_roofline.up1

setenv ADVISOR_XE_2018_DIR /global/common/cori/software/intel/advisor_2018.0.2.537542
```

- Copy file and view results

```bash
cp -r $ADVISOR_XE_2018_DIR/ECP-meeting-tutorial/ .

cd ECP-meeting-tutorial/

advxe-gui stencil.advixeexpz
```

- or use NX: [https://nxcloud01.nersc.gov](https://nxcloud01.nersc.gov)