Erlangen Regional Computing Center

FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG

PRACE Workshop: HPC code optimisation workshop

LIKWID Performance Tools – Hands-On

Thomas Gruber

Dense DP matrix-vector-multiplication

- Common operation in HPC codes

```
!$OMP PARALLEL DO PRIVATE(r)
do c = 1 , SIZE=1000
  tmp=x(c)
  do r = 1 , SIZE
    y(r)=y(r) + A(r,c)* tmp
  enddo
enddo
!$OMP END PARALLEL DO
```

Per inner loop:
- 1 ST
- 2 LD
- 2 FP ops

- But what if our input matrix is symmetric?
  - Let‘s use only the triangular matrix
    - Half the FP ops
    - Loading only half of the matrix

```
!$OMP PARALLEL DO PRIVATE(r)
do c = 1 , SIZE
  do r = 1 , c
    y(r)=y(r) + A(r,c)* x(c)
  enddo
enddo
!$OMP END PARALLEL DO
```

Dense DP matrix-vector-multiplication

```
#pragma omp parallel private(k)
{

  for (k = 0; k < ROUNDS; k++) {
    #pragma omp for private(j)
    for (i = 0; i < N; i++)
    {
      for (j = 0; j < i; ++j) {
        cvec[i] += mat[offset+j]
                 * bvec[j];
      }
    }
  }
}
```

- Compile with `-DLIKWID_PERFMON`
- Link with LIKWID library (`-llikwid`)
- `LIKWID_MARKER_REGISTER()` recommended

```
#include <likwid-marker.h>
LIKWID_MARKER_INIT;
#pragma omp parallel private(k)
{
  LIKWID_MARKER_START("dMVM")
  for (k = 0; k < ROUNDS; k++) {
    #pragma omp for private(j)
    for (i = 0; i < N; i++)
    {
      for (j = 0; j < i; ++j) {
        cvec[i] += mat[offset+j]
                 * bvec[j];
      }
    }
  }
  LIKWID_MARKER_STOP("dMVM")
}
LIKWID_MARKER_CLOSE;
```

Dense DP matrix-vector-multiplication

```
$ likwid-perfctr -C S0:0-2 -g L2 -m ./matrix
--------------------------------------------------------------------------------
CPU name:         Intel(R) Xeon(R) CPU E5-2630 v4 @ 2.20GHz
CPU type:         Intel Xeon Broadwell EN/EP/EX processor
CPU clock:        2.19 GHz
--------------------------------------------------------------------------------
Region MVM, Group 1: L2
+----------------------+---------+------------+------------+------------+
|         Event        | Counter |   Core 0   |   Core 1   |   Core 2   |
+----------------------+---------+------------+------------+------------+
|   INSTR RETIRED ANY  |  FIXC0  | 107645000  | 114450900  | 123952600  |
|  L2D load data volume [GBytes]  |     1.1408 |     0.6951 |     0.2488 |
| L2D evict data volume [GBytes]  |     0.0021 |     0.0032 |     0.0031 |
+----------------------+---------+------------+------------+------------+
```

Number of instructions raise with CPU number

First CPU loads most data into L1

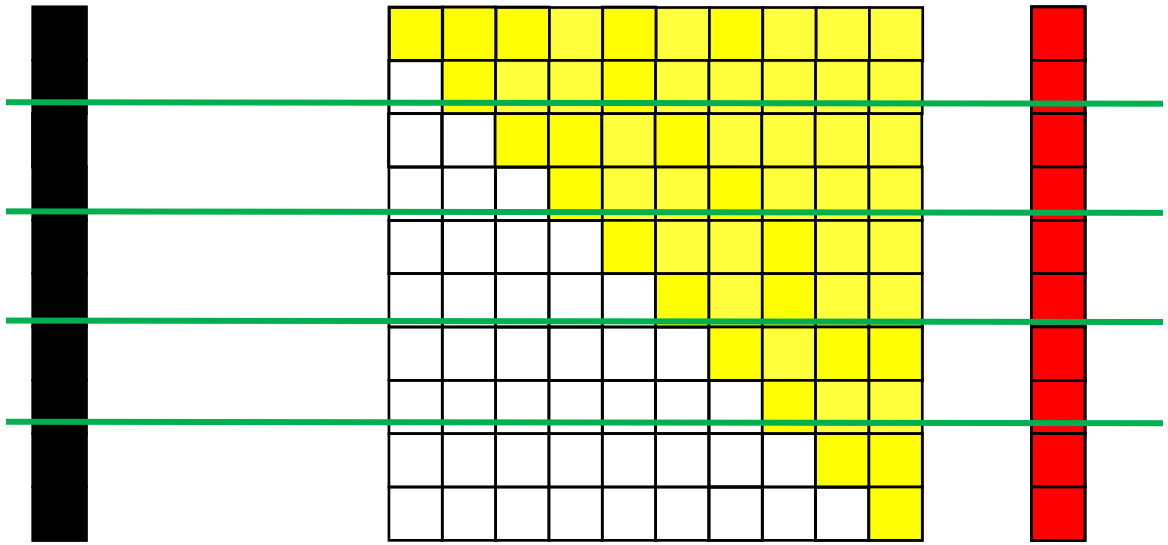
Always use a reasonable metric for "work"! Might not be FLOP/s!

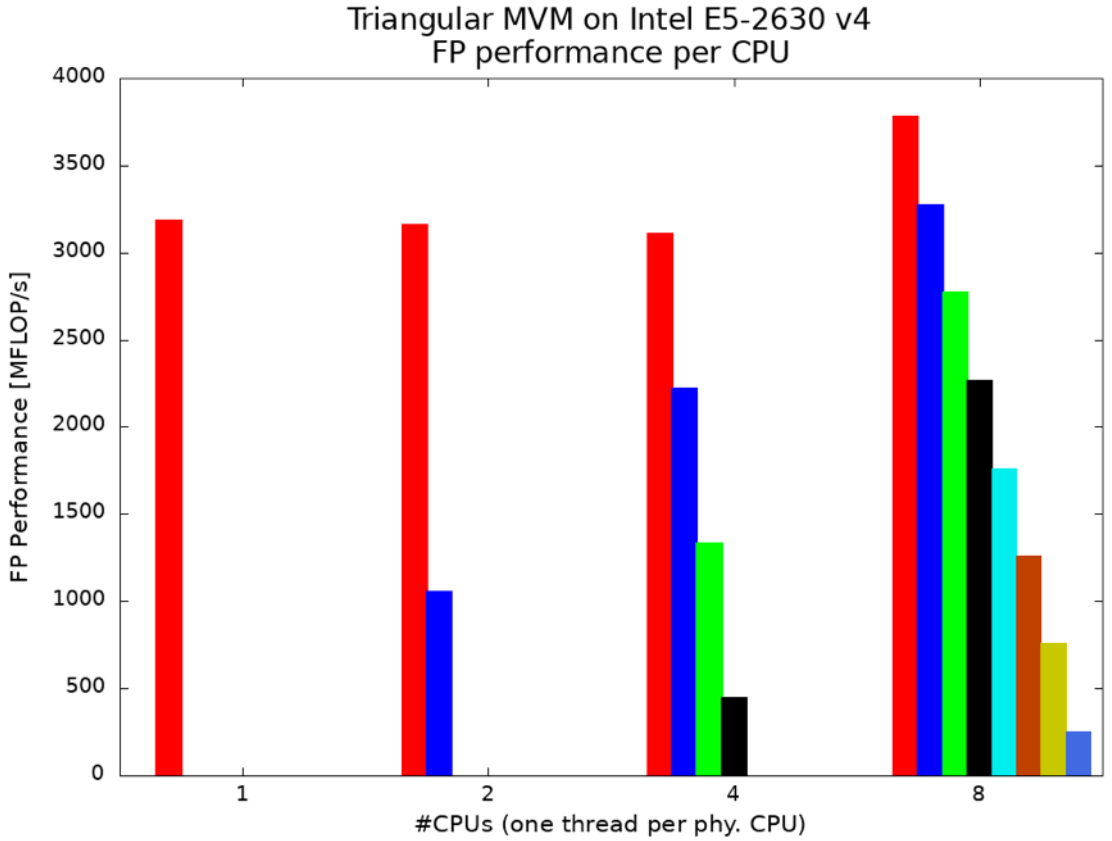
Dense DP matrix-vector-multiplication

- Why are instructions raising?

```
!$OMP PARALLEL DO PRIVATE(r)
do c = 1 , SIZE
  do r = 1 , c
     y(r)=y(r) + A(r,c)* x(c)
  enddo
enddo
!$OMP END PARALLEL DO
```

Implicit OpenMP barrier
(busy waiting then sleep)

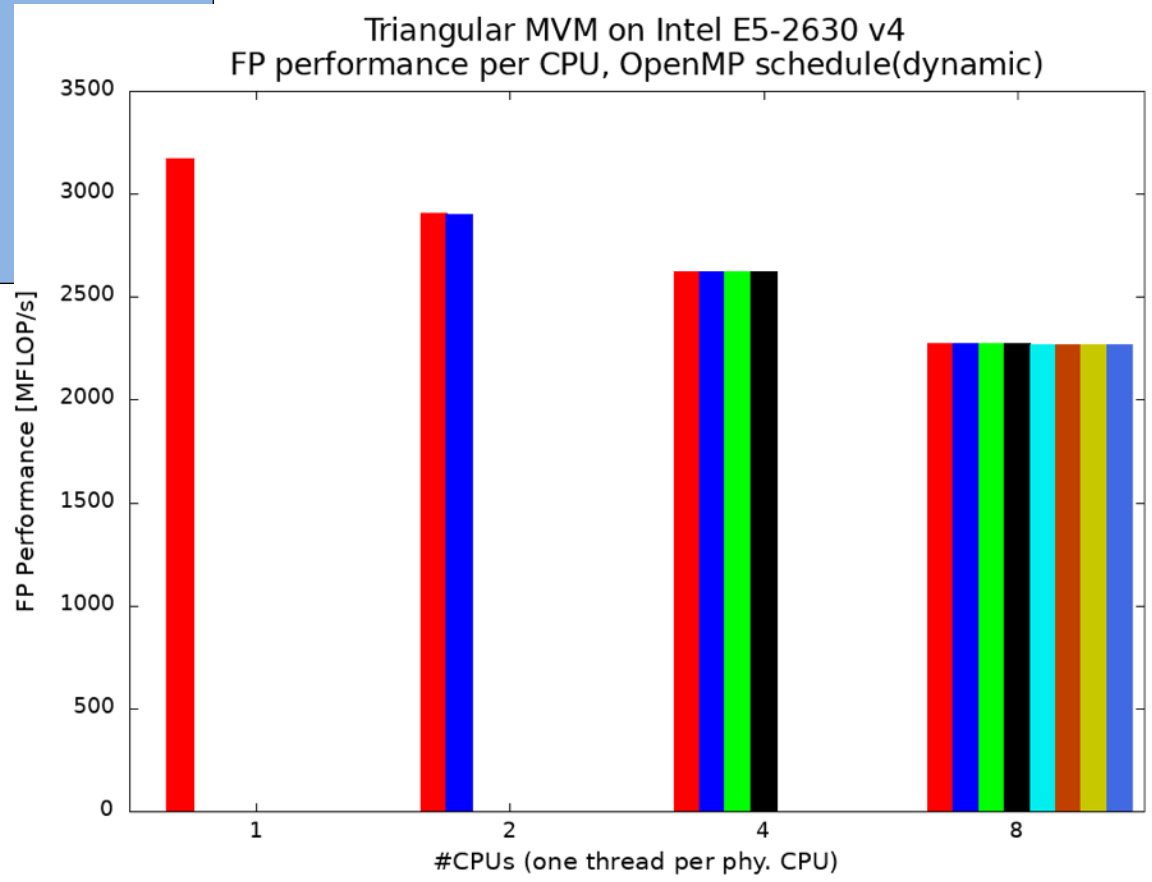




Dense DP matrix-vector-multiplication

- OpenMP schedule to the rescue!

```
!$OMP PARALLEL DO PRIVATE(r) SCHEDULE(DYNAMIC)
do c = 1 , SIZE
  do r = 1 , c
     y(r)=y(r) + A(r,c)* x(c)
  enddo
enddo
!$OMP END PARALLEL DO
```



- Fill wait time with work
- Slower for fully regular problems (dyn. chunk calculation)

Erlangen Regional Computing Center

FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG

Create Roofline Model with LIKWID

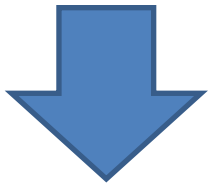
High Performance Computing

Create Roofline Model with LIKWID

- Roofline Model was introduced on Tuesday
- What do we need to create it ourselves?

From the machine:

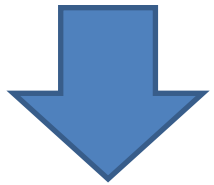
- Peak / Maximum FP rate
- Peak / Maximum bandwidth of X



`likwid-bench`

From our application

- FP rate
- Arithmetic intensity (AI)



`likwid-perfctr`
`likwid-mpirun`

Create Roofline Model with LIKWID

- likwid-bench is a micro-benchmarking tool
- Main interest streaming access patterns
- All benchmarks written in assembly
- Testcases `<kernel>_[sp_]_[mem_]_<vec>[_<extra>]`
  - `<mem>` for non-temporal stores
  - `<vec>` in `sse`, `avx`, `avx512`, `neon`, `vsx`,...
  - `<extra>` commonly `fma`
- Testcases of interest for Roofline:
  `peakflops_*`, `load_*`, `update_*`, `copy_*`

Create Roofline Model with LIKWID

- `$ likwid-bench -t peakflops_avx \`
  `-w N:400kB:40`

  `MFlops/s: 368950.94`

10kB per thread to fit into L1 cache

- `$ likwid-bench -t load_avx \`
  `-W N:4GB:20:1:2`

  `MByte/s: 101789.22`

-W: thread-local initialization
20:1:2: 20 threads w/o SMT

- `$ likwid-perfctr -C S0:0-9@S1:0-9 \`
  `-g MEM_DP -m ./a.out`

  `Operational intensity STAT SUM: 1.3230`
  `MFlops/s STAT SUM: 26312.4563`

S0:0-9@S1:0-9: 10 phys. cores on each socket
MEM_DP: MEM + FLOPS_DP

Create Roofline Model with LIKWID

- Simple gnuplot script:

```
maxflops = 368950.94
maxband = 101789.22
r(x) = maxflops > (x*maxband) ? (x*maxband) : maxflops
ai = 1.3230
mflops = 26312.4563
set obj circ at ai,mflops \
        size 0.1 fs solid
plot r(x)
```