OpenMP®

SC'21 Booth Talk Series

NUMA in OpenMP - Home Sweet Home

Part II - NUMA support in OpenMP

Ruud van der Pas, Oracle Linux Engineering

NUMA Support in OpenMP

NUMA, OpenMP and Thread Affinity

Tuning for NUMA is about keeping threads and their data close

In OpenMP, a thread may be moved to the data

Not the other way round, because that is more expensive

The affinity constructs in OpenMP control where threads execute

This is a powerful feature and should be used when optimizing the application for NUMA

The Affinity Controls in OpenMP 5.1

There are two environment variables to control the affinity

OMP_PLACES defines where threads are allowed to execute

Choices are: threads, cores, ll_caches, numa_domains, sockets, or low level hardware thread IDs

OMP_PROC_BIND controls the mapping of threads onto places

Choices are: true, false, primary, close, or spread

Choices for the abstract OpenMP Places

Keyword	Place definition
threads	*A hardware thread*
cores	*A core*
ll_caches	*A set of cores that share the last level cache*
numa_domain	*A set of cores that share a memory and have the same distance to this memory*
sockets	*A single socket*

An Example

```
# Use 16 threads:
export OMP_NUM_THREADS=16

# Use 2 sockets to place those threads:
export OMP_PLACES=sockets(2)

# Spread them as far apart as possible:
export OMP_PROC_BIND=spread

# Run the code as usual:
./a.out
```

Affinity Controls in the Source Code

Where relevant, the OpenMP directives also support NUMA

They are implemented as a clause

For example, the proc_bind clause on the parallel directive

There are also runtime functions for affinity

An Example Code

```
 1 int main(int argc, char **argv)
 2 {
 3  #pragma omp parallel
 4  {
 5    int TID       = omp_get_thread_num();
 6    int place_num = omp_get_place_num();
 7    printf("Hello World from Thread %d on Place Number %d\n",
 8           TID, place_num);
 9  } // End of parallel region
10  return(0);
11 }
```

An Example Code

```
$ gcc -fopenmp hello_world_omp.c
$ export OMP_NUM_THREADS=4
$ export OMP_PLACES=cores
$ export OMP_PROC_BIND=spread
$ ./a.out
Hello World from Thread 0 on Place Number 0
Hello World from Thread 2 on Place Number 4
Hello World from Thread 1 on Place Number 2
Hello World from Thread 3 on Place Number 6
$
```

Where are these place numbers in the system?

Diagnostics

Mistakes in the NUMA setup may go unnoticed

The diagnostic features come in very handy!

In particular the following environment variables:

Variable	Functionality
OMP_DISPLAY_ENV	*Echo the settings of OpenMP environment variables*
OMP_DISPLAY_AFFINITY	*Display runtime affinity information*

The Example Revisited

```
$ export OMP_NUM_THREADS=4
$ export OMP_PLACES=cores
$ export OMP_PROC_BIND=spread
$ export OMP_DISPLAY_ENV=verbose
$ ./a.out

OPENMP DISPLAY ENVIRONMENT BEGIN
  _OPENMP = '201511'
  OMP_DYNAMIC = 'FALSE'
  OMP_NESTED = 'FALSE'
  OMP_NUM_THREADS = '4'
  OMP_SCHEDULE = 'DYNAMIC'
   OMP_PROC_BIND = 'SPREAD'
   OMP_PLACES = '{0:2},{2:2},{4:2},{6:2},{8:2},{10:2},{12:2},{14:2}'
  OMP_STACKSIZE = '140737506661931'
  OMP_WAIT_POLICY = 'PASSIVE'
  OMP_THREAD_LIMIT = '4294967295'
  OMP_MAX_ACTIVE_LEVELS = '2147483647'
  OMP_CANCELLATION = 'FALSE'
        … etc …
OPENMP DISPLAY ENVIRONMENT END
```

```
Hello World from Thread 3 on Place Number 6
Hello World from Thread 0 on Place Number 0
Hello World from Thread 2 on Place Number 4
Hello World from Thread 1 on Place Number 2
```

Much More to Explore

Additional Topics Related to NUMA

It is beyond the scope of this talk to go into the details, but much more support for NUMA is available in 5.1

A rich set of tailored memory allocators are supported for example

These allow the memory behaviour to be customized and tuned based upon the application needs

Section 2.13.3 about the allocate directive is a good starting point

Wrapping Things Up

Wrapping Things Up

Hopefully this talk has helped to show what OpenMP provides to help optimizing an application for a NUMA architecture

But there is more and it is worth exploring to see what else is available in OpenMP to tune for a NUMA system

Thank You And … Stay Tuned!

Bad OpenMP
Does Not Scale

Ruud van der Pas

SC'21 OpenMP Booth Talk

OpenMP®

SC'21 Booth Talk Series

openmp.org OpenMP API specs, forum, reference guides, and more

link.openmp.org/sc21 Videos and PDFs of OpenMP SC'21 presentations