VSB TECHNICAL UNIVERSITY OF OSTRAVA | IT4INNOVATIONS NATIONAL SUPERCOMPUTING CENTER

INTRODUCTION TO HIGH PERFORMANCE COMPUTING

PARALLEL PROGRAMMING BASICS

Ondřej Meca

PROGRAMMING PARALLEL MACHINES

Sequential (serial) programs:

- operate on similar principles everywhere

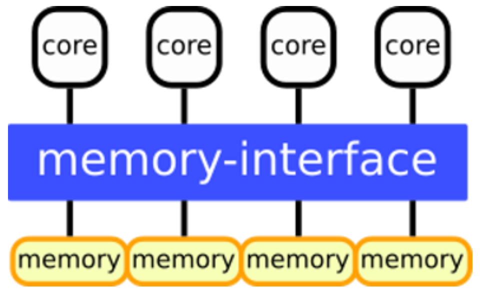
Parallel programs:

- cannot be created just by formal changes of the sequential variant
- can be qualitatively different from the corresponding sequential ones
- dependent on the target parallel architecture
- more difficult to write than sequential ones
    - several new classes of software bugs (e.g., race conditions)
    - difficult debugging
    - issues of scalability...

PARALLEL HARDWARE

Multi-processor (socket)

- all cores share the same memory
- single / global address space
- the same speed to all memory locations (uniform memory access)



socket

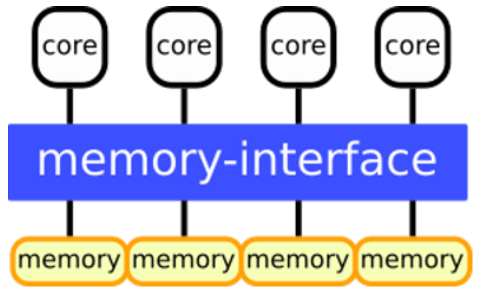
UMA (uniform memory access)

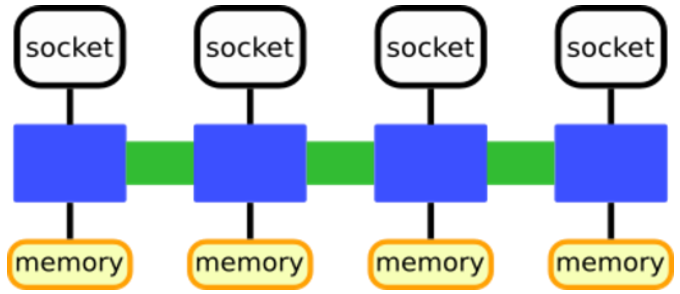
SMP (symmetric multi-processing)

PARALLEL HARDWARE

Several sockets with multi-processors (node)

- memory is shared among all CPUs
- single / global address space
- the same speed to all memory locations (uniform memory access)?





socket

UMA (uniform memory access)

SMP (symmetric multi-processing)

node

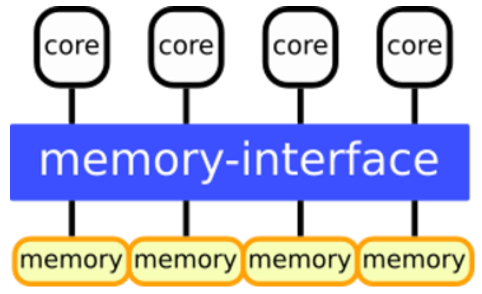
ccNUMA (cache-coherent non-uniform ...)

first touch, pinning!

PARALLEL HARDWARE

Several sockets with multi-processors (node)

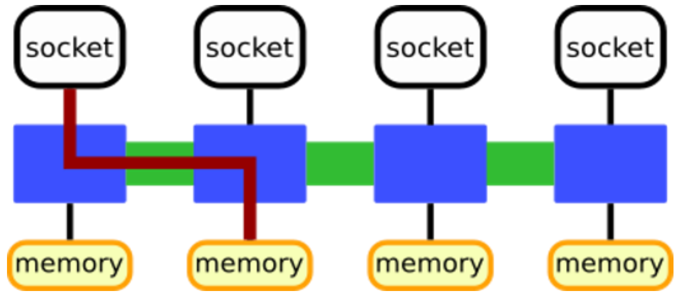
- memory is shared among all CPUs
- single / global address space
- ~~the same speed to all memory locations (uniform memory access)?~~
- the speed is dependent on a memory location (non-uniform memory access)



socket

UMA (uniform memory access)

SMP (symmetric multi-processing)



node

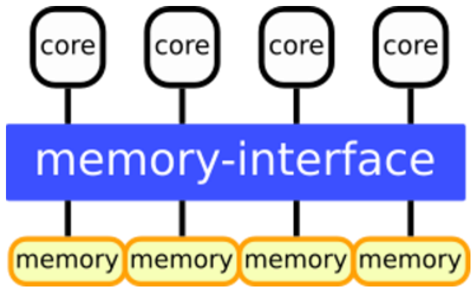
ccNUMA (cache-coherent non-uniform ...)

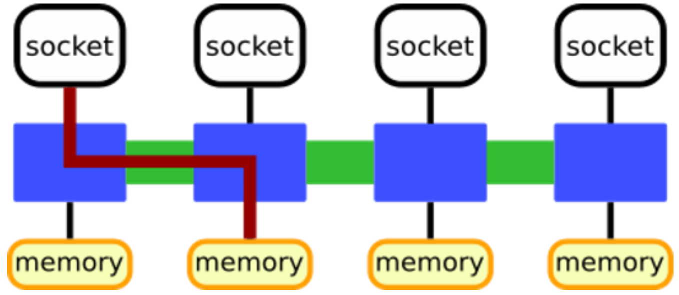
first touch, pinning!

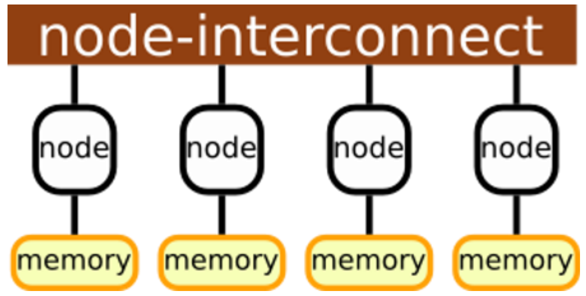
PARALLEL HARDWARE

Multi-computers with various architectures (cluster)

- set of nodes interconnected by a network
- each node has separated memory
- slower access to memories of other processors
- accelerated nodes







socket

UMA (uniform memory access)

SMP (symmetric multi-processing)

node

ccNUMA (cache-coherent non-uniform ...)

first touch, pinning!

cluster

NUMA (non-uniform memory access)

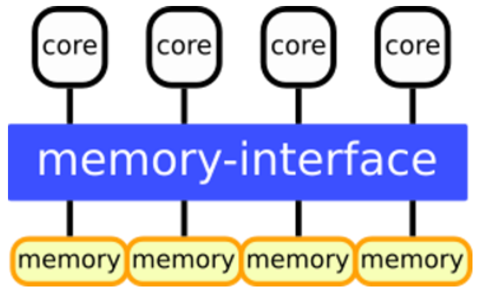
fast access to own memory only

PARALLEL HARDWARE

OpenMP: shared memory (socket, node)

MPI: distributed memory (socket, node, cluster)

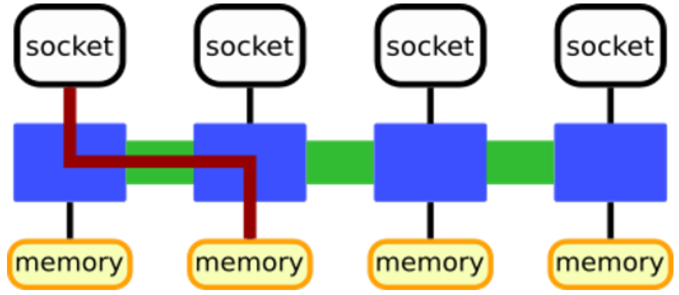
CUDA: accelerated nodes



socket

UMA (uniform memory access)

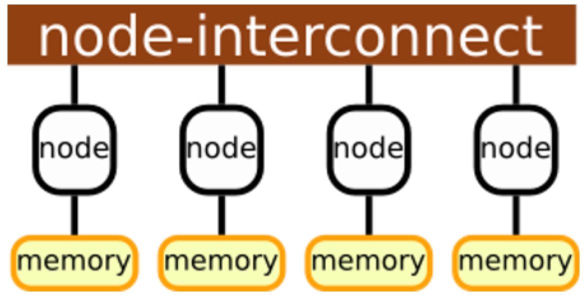
SMP (symmetric multi-processing)



node

ccNUMA (cache-coherent non-uniform ...)

first touch, pinning!



cluster

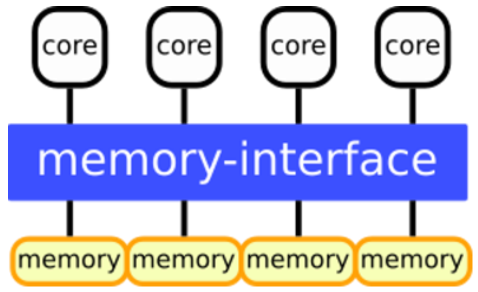
NUMA (non-uniform memory access)

fast access to own memory only

PARALLEL HARDWARE

Hybrid approach

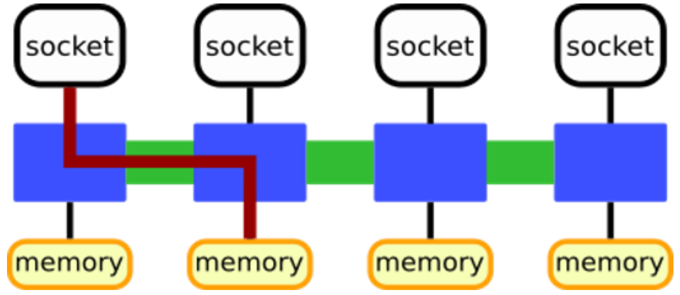
- combination of more approaches (OpenMP, MPI, CUDA,...)
- potential to fully utilize current (future) hardware



socket

UMA (uniform memory access)

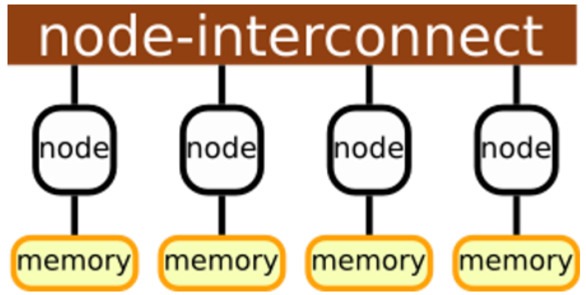
SMP (symmetric multi-processing)



node

ccNUMA (cache-coherent non-uniform ...)

first touch, pinning!



cluster

NUMA (non-uniform memory access)

fast access to own memory only

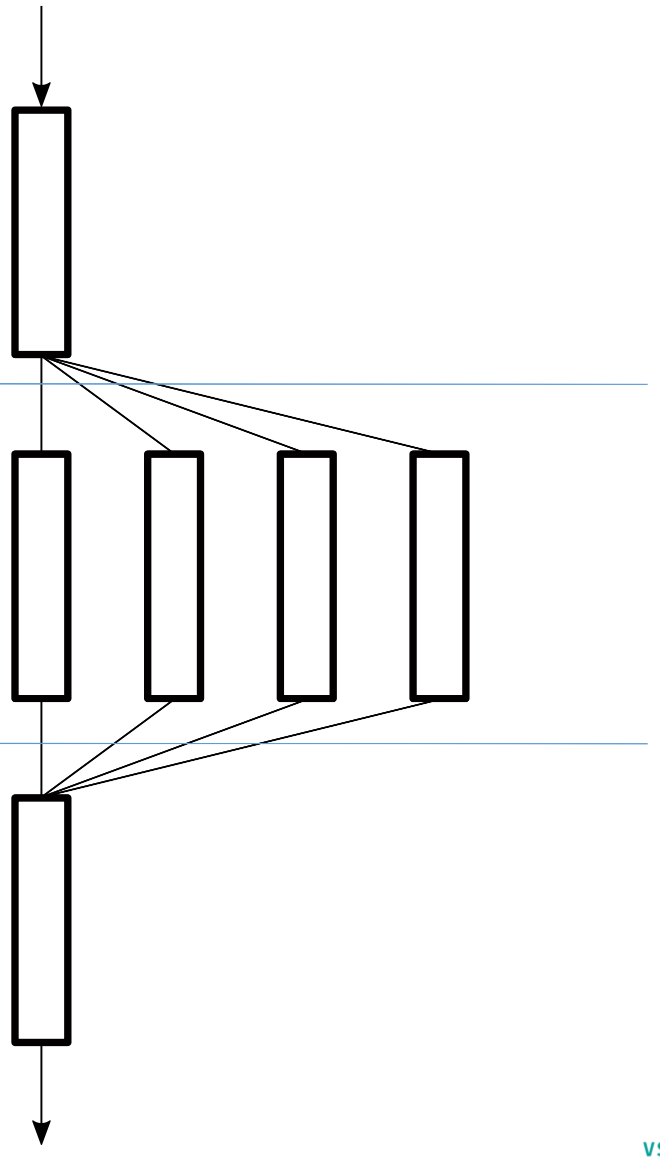
OPENMP

- **Open M**ulti-**P**rocessing
  - API for writing portable multi-threaded applications based on the shared variables model with interfaces for Fortran, C, and C++
  - compilers available on most platforms (Unix, Windows, etc.)
- A set of compiler directives, library routines and environment variables
- A standard developed by the OpenMP Architecture Review Board
  - http://www.openmp.org
  - first specification in 1997, current version 5.2

- No data distribution, no communication (threads communicate via shared variables)
- Allows incremental parallelization
  - i.e., the sequential program evolves into a parallel program
  - single source code for both the sequential and parallel versions

OPENMP

```
#include "omp.h"

int main(int argc, char **argv) {
  int iam = 0, np = 1;

  #pragma omp parallel private(iam, np) /* Parallel region */
  {
    #if defined (_OPENMP)
      np = omp_get_num_threads();
      iam = omp_get_thread_num();
    #endif
    printf("Hello from thread %d out of %d\n", iam, np);
  }
}
```

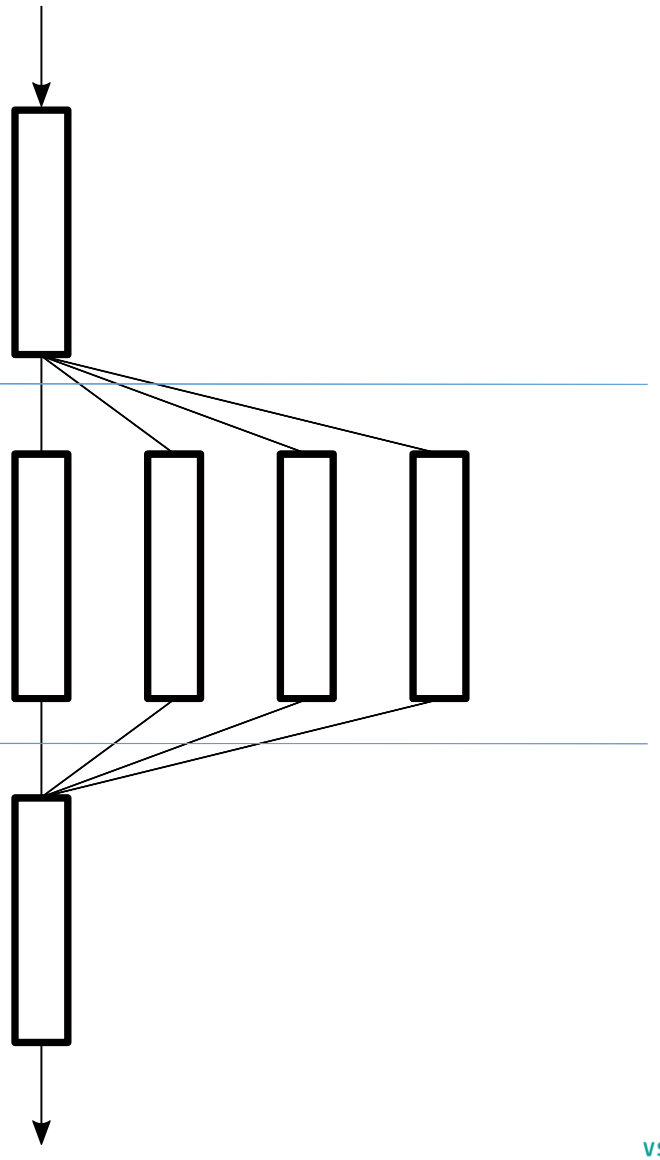


```
$ g++ -fopenmp hello.cpp -o hello
$ OMP_NUM_THREADS=4 ./hello

Hello from thread 2 out of 4
Hello from thread 0 out of 4
Hello from thread 1 out of 4
Hello from thread 3 out of 4
```

OPENMP

```
#include "omp.h"

int main(int argc, char **argv) {
  int iam = 0, np = 1;

  #pragma omp parallel private(iam, np) /* Parallel region */
  {
    #if defined (_OPENMP)
      np = omp_get_num_threads();
      iam = omp_get_thread_num();
    #endif
    printf("Hello from thread %d out of %d\n", iam, np);
  }
}
```

```
$ g++ -fopenmp hello.cpp -o hello
$ OMP_NUM_THREADS=4 ./hello

Hello from thread 2 out of 4
Hello from thread 0 out of 4
Hello from thread 1 out of 4
Hello from thread 3 out of 4
```



OPENMP

- Mainly directives applied to the following block of code

```
#pragma omp parallel [ clause [ [ , ] clause ] ... ] new-line
{
    // code performed by all threads
}
```

- Clauses:
  - private (list), shared (list),
  - reduction (operator: list), schedule (type [, chunk])
- Synchronization:
  - master, critical, atomic, barrier
- Environment variables:
  - OMP_NUM_THREADS, OMP_PLACES, OMP_PROC_BIND

- **https://www.openmp.org/resources/tutorials-articles/**
- https://pages.tacc.utexas.edu/~eijkhout/pcse/html/omp-affinity.html

OPENMP

- Mainly directives applied to the following block of code

```
#pragma omp parallel [ clause [ [ , ] clause ] ... ] new-line
{
        // code performed by all threads
}
```

- Clauses:
  - private (list), shared (list),
  - reduction (operator: list), schedule (type [, chunk])
- Synchronization:
  - master, critical, atomic, barrier
- **Environment variables:**
  - **OMP_NUM_THREADS, OMP_PLACES, OMP_PROC_BIND**

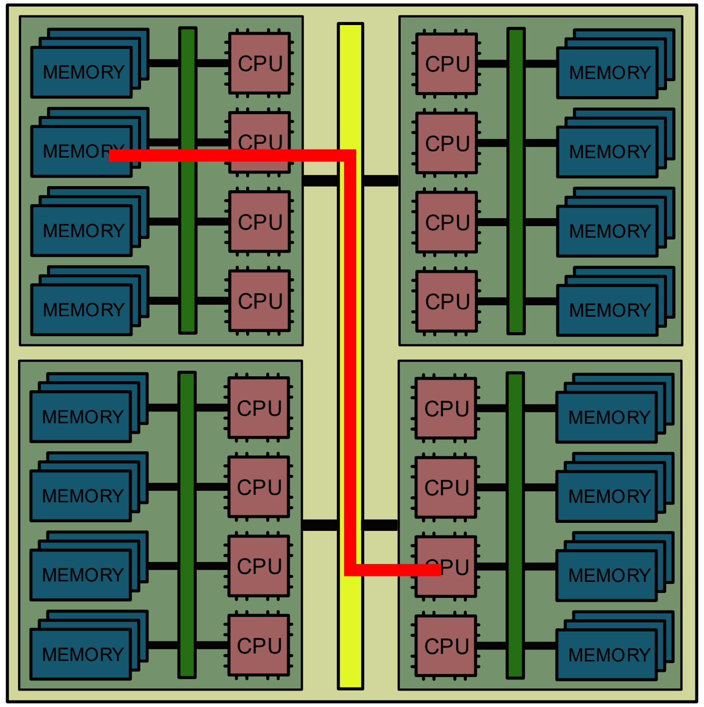
- **https://www.openmp.org/resources/tutorials-articles/**
- https://pages.tacc.utexas.edu/~eijkhout/pcse/html/omp-affinity.html

OPENMP PITFALLS

- Race conditions
  - output is dependent on the detailed timing of concurrent operations
  - e.g., modifying the same variable by two threads
- Deadlocks
  - waiting for resources that will never be available
- Sequential equivalence:
  - strong: bitwise identical results
  - weak: mathematically equivalent (not bitwise identical due to the floating-point arithmetic)

OPENMP PITFALLS

- Cache coherent distributed memory (ccNUMA)
  - threads requests memory that was firstly touched by a thread from another sockets
  - the same memory should be accessed by the same thread
  - fix threads to a particular CPUs (OMP_PROC_BIND=true ./app)



```
double *vals = new double[rows * cols];

#pragma omp parallel for collapse(2)
for (int r = 0; r < rows; ++r) {
    for (int c = 0; c < cols; ++c) {
        vals[r * cols + c] = 0;
    }
}
```

MPI

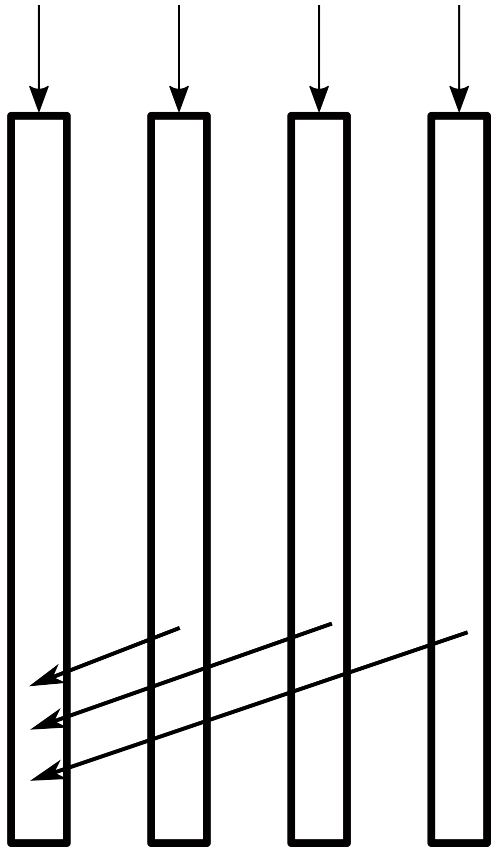
- **Message Passing Interface:**
  - standard for distributed memory parallelism with passing messages
- MPI is the interface, not a library!
  - many available libraries with an implementation (OpenMPI, mpich, Intel MPI,...)
  - some behavior is dependent on a particular implementation
- A standard developed by the MPI Forum
  - http://www.mpi-forum.org
  - first specification in 1994, current version 4.0
- Explicit definition of data distribution and communication
- MPI application is a set of processes that cooperate with each other by sending messages

MPI

```
#include "mpi.h"

int main(int argc, char **argv) {
  int rank, size;
  MPI_Init(&argc, &argv);
  MPI_Comm_rank(MPI_COMM_WORLD, &rank);
  MPI_Comm_size(MPI_COMM_WORLD, &size);

  printf("Hello from process %d out of %d\n", rank, size);
  if (rank==0) {
    // recv messages
  } else {
    // send a message
  }
  MPI_Finalize();
}
```

```
$ mpic++ hello.cpp -o hello
$ mpirun -n 4 ./hello
```

```
Hello from process 2 out of 4
Hello from process 0 out of 4
Hello from process 1 out of 4
Hello from process 3 out of 4
```



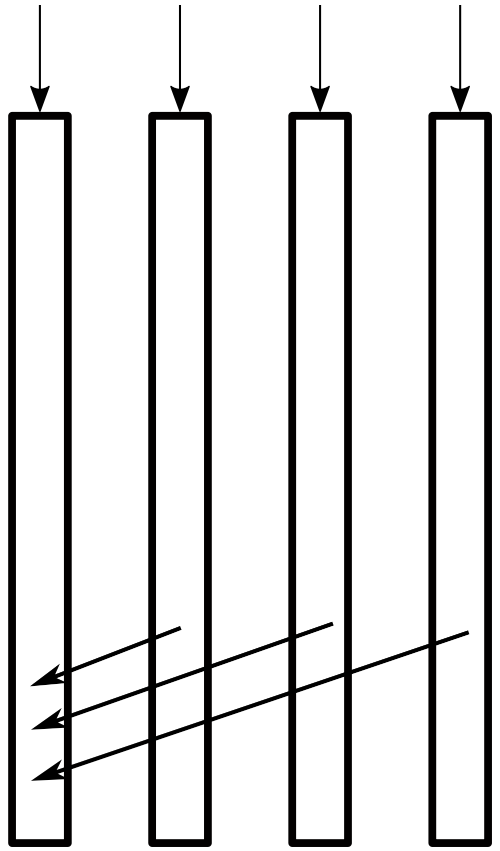
VSB TECHNICAL UNIVERSITY OF OSTRAVA | IT4INNOVATIONS NATIONAL SUPERCOMPUTING CENTER

MPI

```
#include "mpi.h"

int main(int argc, char **argv) {
  int rank, size;
  MPI_Init(&argc, &argv);
  MPI_Comm_rank(MPI_COMM_WORLD, &rank);
  MPI_Comm_size(MPI_COMM_WORLD, &size);

  printf("Hello from process %d out of %d\n", rank, size);
  if (rank==0) {
    // recv messages
  } else {
    // send a message
  }
  MPI_Finalize();
}
```

```
$ mpic++ hello.cpp -o hello
$ mpirun -n 4 ./hello
```

```
Hello from process 2 out of 4
Hello from process 0 out of 4
Hello from process 1 out of 4
Hello from process 3 out of 4
```



MPI PITFALLS

- Race conditions
  - output is dependent on the detailed timing sent/received operations
  - e.g., output is dependent on the order of received messages
- Deadlocks
  - waiting for resources that will never be available
- Sequential equivalence:
  - strong: bitwise identical results
  - weak: mathematically equivalent (not bitwise identical due to the floating-points arithmetic)

MPI PITFALLS

- Non-scalable functions / patterns
  - collectives with input of size *O(#processes)*

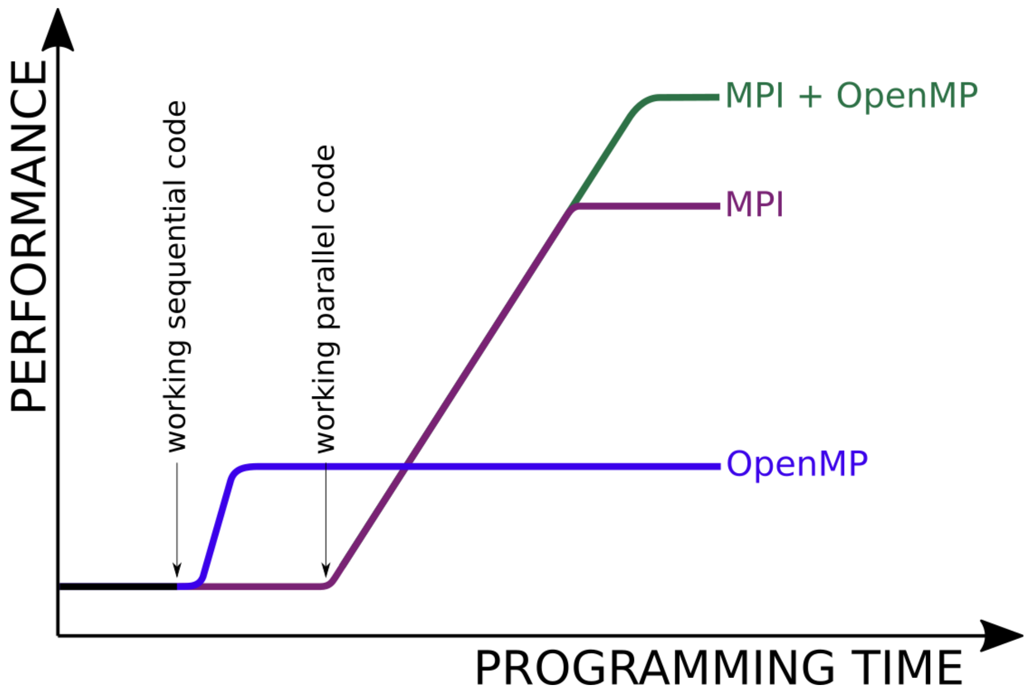
- Serialization:
  - the order of messages serializes the application
  - e.g., each process must wait to a message from the previous process

- Expensive communication
  - exchanging too much of data

- Performance is not portable
  - MPI assures only portable application!

OPENMP VS. MPI

- OpenMP
  - Incremental parallelization
- MPI:
  - usually new application with potential to fully utilize cluster capacities



PARALLEL RUN

How to run your parallel application?

PARALLEL RUN

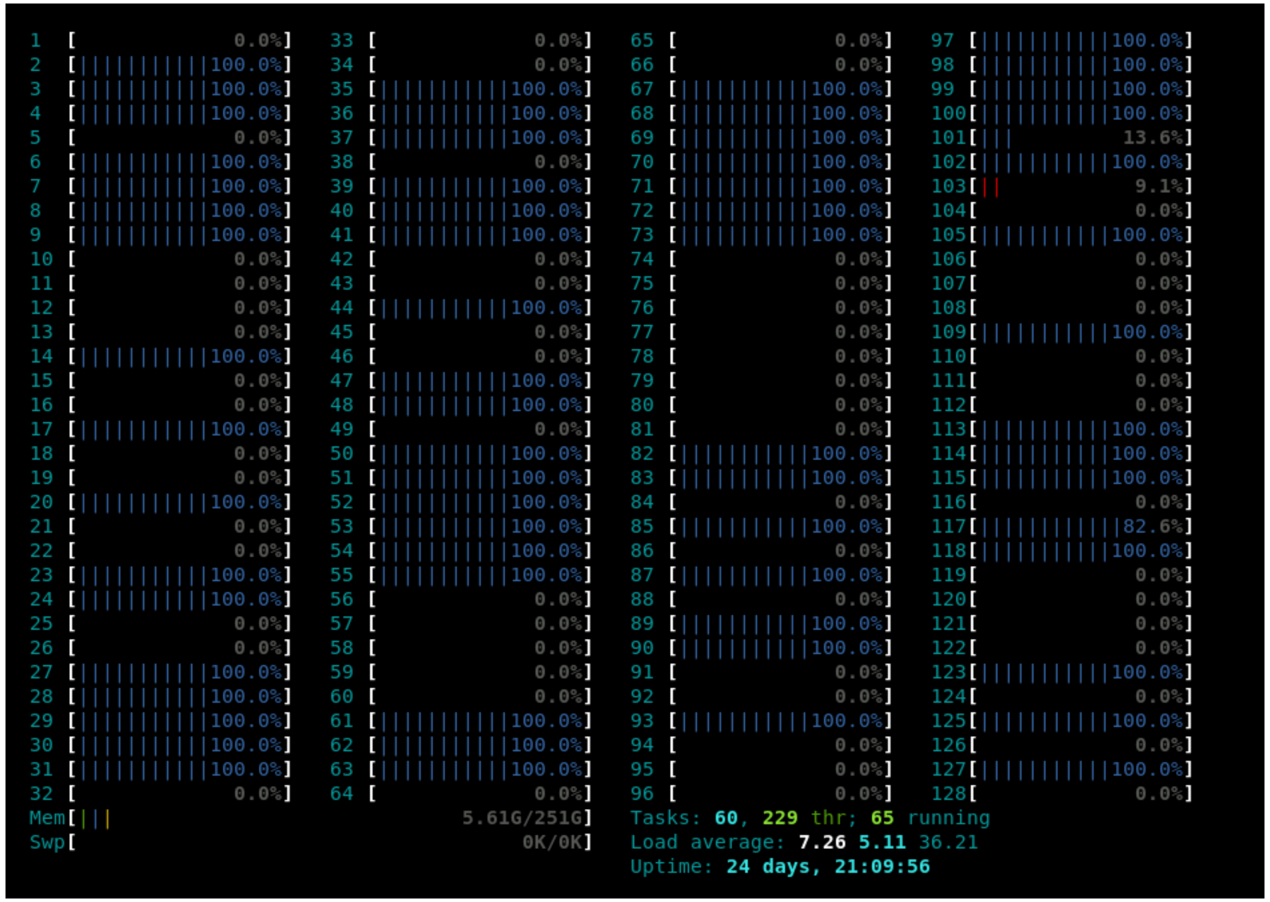
- PBS settings
  - https://docs.it4i.cz/general/job-submission-and-execution/
  - set correct number of MPI processes and OMP threads
    - qsub -l select=2:ncpus=128:**mpiprocs**=8:**ompthreads**=16
  - **mpiprocs**: number of MPI processes per node
  - **ompthreads**: number of OMP threads per MPI process

```
$ qsub -ADD-22-46 -qqprod -lselect=2:ncpus=128:mpiprocs=2:ompthreads=64 -I
qsub: waiting for job 1219022.infra-pbs to start
qsub: job 1219022.infra-pbs ready

$ echo $PBS_NODEFILE
/var/spool/pbs/aux/1219022.infra-pbs

$ cat /var/spool/pbs/aux/1219022.infra-pbs
cn140.karolina.it4i.cz
cn140.karolina.it4i.cz
cn141.karolina.it4i.cz
cn141.karolina.it4i.cz

$ echo $OMP_NUM_THREADS
64
```

PARALLEL RUN

```
$ qsub -ADD-22-46 -qqprod -lselect=2:ncpus=128:mpiprocs=2:ompthreads=64 -I
$ mpirun -n 1 ./threaded
```

```
$ ssh cnXXX
$ htop -d2
```



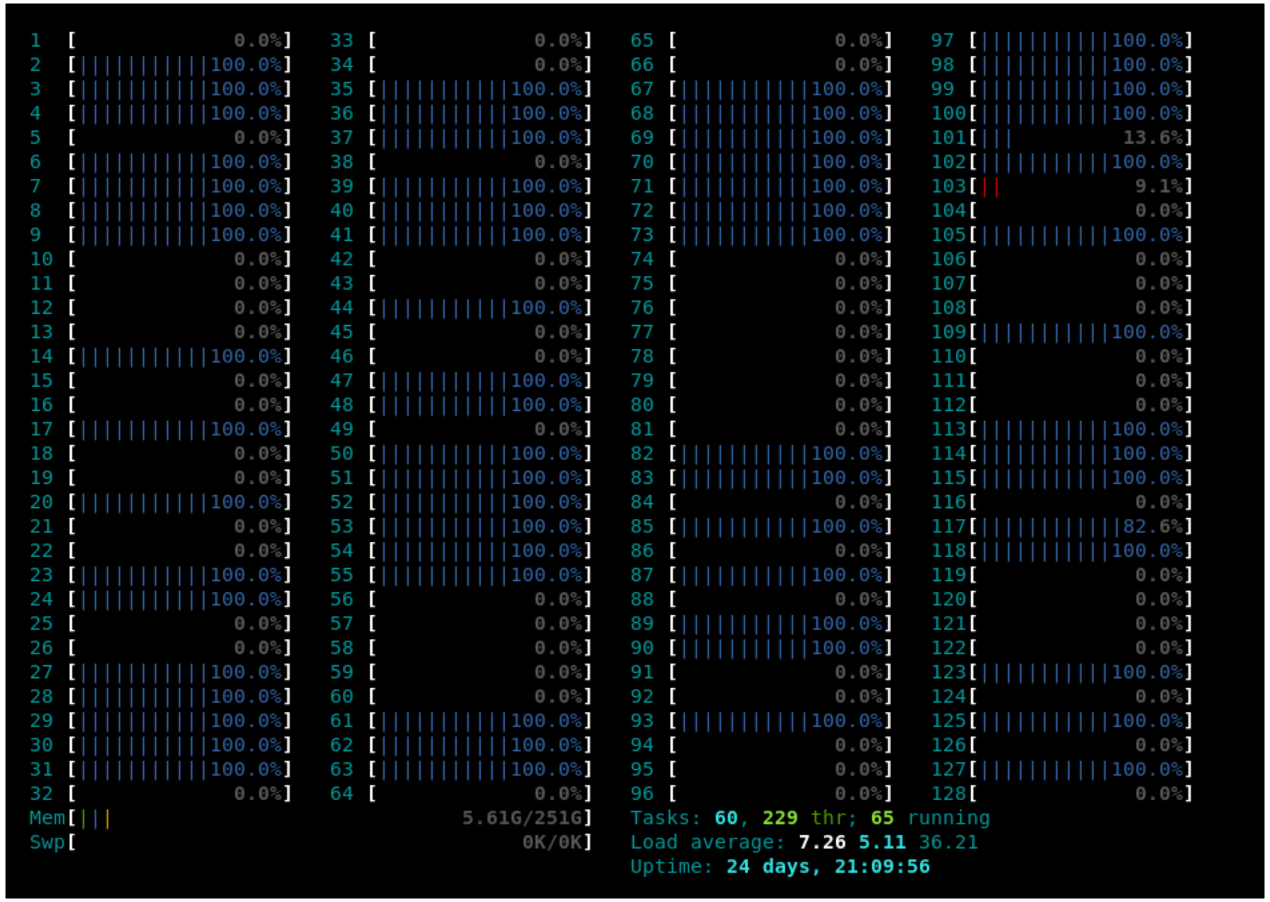
```
1  [            0.0%]   33 [            0.0%]   65 [            0.0%]   97 [||||||||||100.0%]
2  [||||||||||100.0%]   34 [            0.0%]   66 [            0.0%]   98 [||||||||||100.0%]
3  [||||||||||100.0%]   35 [||||||||||100.0%]   67 [||||||||||100.0%]   99 [||||||||||100.0%]
4  [||||||||||100.0%]   36 [||||||||||100.0%]   68 [||||||||||100.0%]   100[||||||||||100.0%]
5  [            0.0%]   37 [||||||||||100.0%]   69 [||||||||||100.0%]   101[|||        13.6%]
6  [||||||||||100.0%]   38 [            0.0%]   70 [||||||||||100.0%]   102[||||||||||100.0%]
7  [||||||||||100.0%]   39 [||||||||||100.0%]   71 [||||||||||100.0%]   103[||          9.1%]
8  [||||||||||100.0%]   40 [||||||||||100.0%]   72 [||||||||||100.0%]   104[            0.0%]
9  [||||||||||100.0%]   41 [||||||||||100.0%]   73 [||||||||||100.0%]   105[||||||||||100.0%]
10 [            0.0%]   42 [            0.0%]   74 [            0.0%]   106[            0.0%]
11 [            0.0%]   43 [            0.0%]   75 [            0.0%]   107[            0.0%]
12 [            0.0%]   44 [||||||||||100.0%]   76 [            0.0%]   108[            0.0%]
13 [            0.0%]   45 [            0.0%]   77 [            0.0%]   109[||||||||||100.0%]
14 [||||||||||100.0%]   46 [            0.0%]   78 [            0.0%]   110[            0.0%]
15 [            0.0%]   47 [||||||||||100.0%]   79 [            0.0%]   111[            0.0%]
16 [            0.0%]   48 [||||||||||100.0%]   80 [            0.0%]   112[            0.0%]
17 [||||||||||100.0%]   49 [            0.0%]   81 [            0.0%]   113[||||||||||100.0%]
18 [            0.0%]   50 [||||||||||100.0%]   82 [||||||||||100.0%]   114[||||||||||100.0%]
19 [            0.0%]   51 [||||||||||100.0%]   83 [||||||||||100.0%]   115[||||||||||100.0%]
20 [||||||||||100.0%]   52 [||||||||||100.0%]   84 [            0.0%]   116[            0.0%]
21 [            0.0%]   53 [||||||||||100.0%]   85 [||||||||||100.0%]   117[|||||||||||82.6%]
22 [            0.0%]   54 [||||||||||100.0%]   86 [            0.0%]   118[||||||||||100.0%]
23 [||||||||||100.0%]   55 [||||||||||100.0%]   87 [||||||||||100.0%]   119[            0.0%]
24 [||||||||||100.0%]   56 [            0.0%]   88 [            0.0%]   120[            0.0%]
25 [            0.0%]   57 [            0.0%]   89 [||||||||||100.0%]   121[            0.0%]
26 [            0.0%]   58 [            0.0%]   90 [||||||||||100.0%]   122[            0.0%]
27 [||||||||||100.0%]   59 [            0.0%]   91 [            0.0%]   123[||||||||||100.0%]
28 [||||||||||100.0%]   60 [            0.0%]   92 [            0.0%]   124[            0.0%]
29 [||||||||||100.0%]   61 [||||||||||100.0%]   93 [||||||||||100.0%]   125[||||||||||100.0%]
30 [||||||||||100.0%]   62 [||||||||||100.0%]   94 [            0.0%]   126[            0.0%]
31 [||||||||||100.0%]   63 [||||||||||100.0%]   95 [            0.0%]   127[||||||||||100.0%]
32 [            0.0%]   64 [            0.0%]   96 [            0.0%]   128[            0.0%]
Mem[|||                           5.61G/251G]   Tasks: 60, 229 thr; 65 running
Swp[                                   0K/0K]   Load average: 7.26 5.11 36.21
                                                Uptime: 24 days, 21:09:56
```

VSB TECHNICAL UNIVERSITY OF OSTRAVA | IT4INNOVATIONS NATIONAL SUPERCOMPUTING CENTER

PARALLEL RUN

How are threads pinned?

How will MPI be pinned?



```
1  [            0.0%]   33 [            0.0%]   65 [            0.0%]   97 [|||||||||||100.0%]
2  [|||||||||||100.0%]   34 [            0.0%]   66 [            0.0%]   98 [|||||||||||100.0%]
3  [|||||||||||100.0%]   35 [|||||||||||100.0%]   67 [|||||||||||100.0%]   99 [|||||||||||100.0%]
4  [|||||||||||100.0%]   36 [|||||||||||100.0%]   68 [|||||||||||100.0%]   100[|||||||||||100.0%]
5  [            0.0%]   37 [|||||||||||100.0%]   69 [|||||||||||100.0%]   101[|||        13.6%]
6  [|||||||||||100.0%]   38 [            0.0%]   70 [|||||||||||100.0%]   102[|||||||||||100.0%]
7  [|||||||||||100.0%]   39 [|||||||||||100.0%]   71 [|||||||||||100.0%]   103[||          9.1%]
8  [|||||||||||100.0%]   40 [|||||||||||100.0%]   72 [|||||||||||100.0%]   104[            0.0%]
9  [|||||||||||100.0%]   41 [|||||||||||100.0%]   73 [|||||||||||100.0%]   105[|||||||||||100.0%]
10 [            0.0%]   42 [            0.0%]   74 [            0.0%]   106[            0.0%]
11 [            0.0%]   43 [            0.0%]   75 [            0.0%]   107[            0.0%]
12 [            0.0%]   44 [|||||||||||100.0%]   76 [            0.0%]   108[            0.0%]
13 [            0.0%]   45 [            0.0%]   77 [            0.0%]   109[|||||||||||100.0%]
14 [|||||||||||100.0%]   46 [            0.0%]   78 [            0.0%]   110[            0.0%]
15 [            0.0%]   47 [|||||||||||100.0%]   79 [            0.0%]   111[            0.0%]
16 [            0.0%]   48 [|||||||||||100.0%]   80 [            0.0%]   112[            0.0%]
17 [|||||||||||100.0%]   49 [            0.0%]   81 [            0.0%]   113[|||||||||||100.0%]
18 [            0.0%]   50 [|||||||||||100.0%]   82 [|||||||||||100.0%]   114[|||||||||||100.0%]
19 [            0.0%]   51 [|||||||||||100.0%]   83 [|||||||||||100.0%]   115[|||||||||||100.0%]
20 [|||||||||||100.0%]   52 [|||||||||||100.0%]   84 [            0.0%]   116[            0.0%]
21 [            0.0%]   53 [|||||||||||100.0%]   85 [|||||||||||100.0%]   117[||||||||||||82.6%]
22 [            0.0%]   54 [|||||||||||100.0%]   86 [            0.0%]   118[|||||||||||100.0%]
23 [|||||||||||100.0%]   55 [|||||||||||100.0%]   87 [|||||||||||100.0%]   119[            0.0%]
24 [|||||||||||100.0%]   56 [            0.0%]   88 [            0.0%]   120[            0.0%]
25 [            0.0%]   57 [            0.0%]   89 [|||||||||||100.0%]   121[            0.0%]
26 [            0.0%]   58 [            0.0%]   90 [|||||||||||100.0%]   122[            0.0%]
27 [|||||||||||100.0%]   59 [            0.0%]   91 [            0.0%]   123[|||||||||||100.0%]
28 [|||||||||||100.0%]   60 [            0.0%]   92 [            0.0%]   124[            0.0%]
29 [|||||||||||100.0%]   61 [|||||||||||100.0%]   93 [|||||||||||100.0%]   125[|||||||||||100.0%]
30 [|||||||||||100.0%]   62 [|||||||||||100.0%]   94 [            0.0%]   126[            0.0%]
31 [|||||||||||100.0%]   63 [|||||||||||100.0%]   95 [            0.0%]   127[|||||||||||100.0%]
32 [            0.0%]   64 [            0.0%]   96 [            0.0%]   128[            0.0%]
Mem[|||                                  5.61G/251G]   Tasks: 60, 229 thr; 65 running
Swp[                                          0K/0K]   Load average: 7.26 5.11 36.21
                                                       Uptime: 24 days, 21:09:56
```

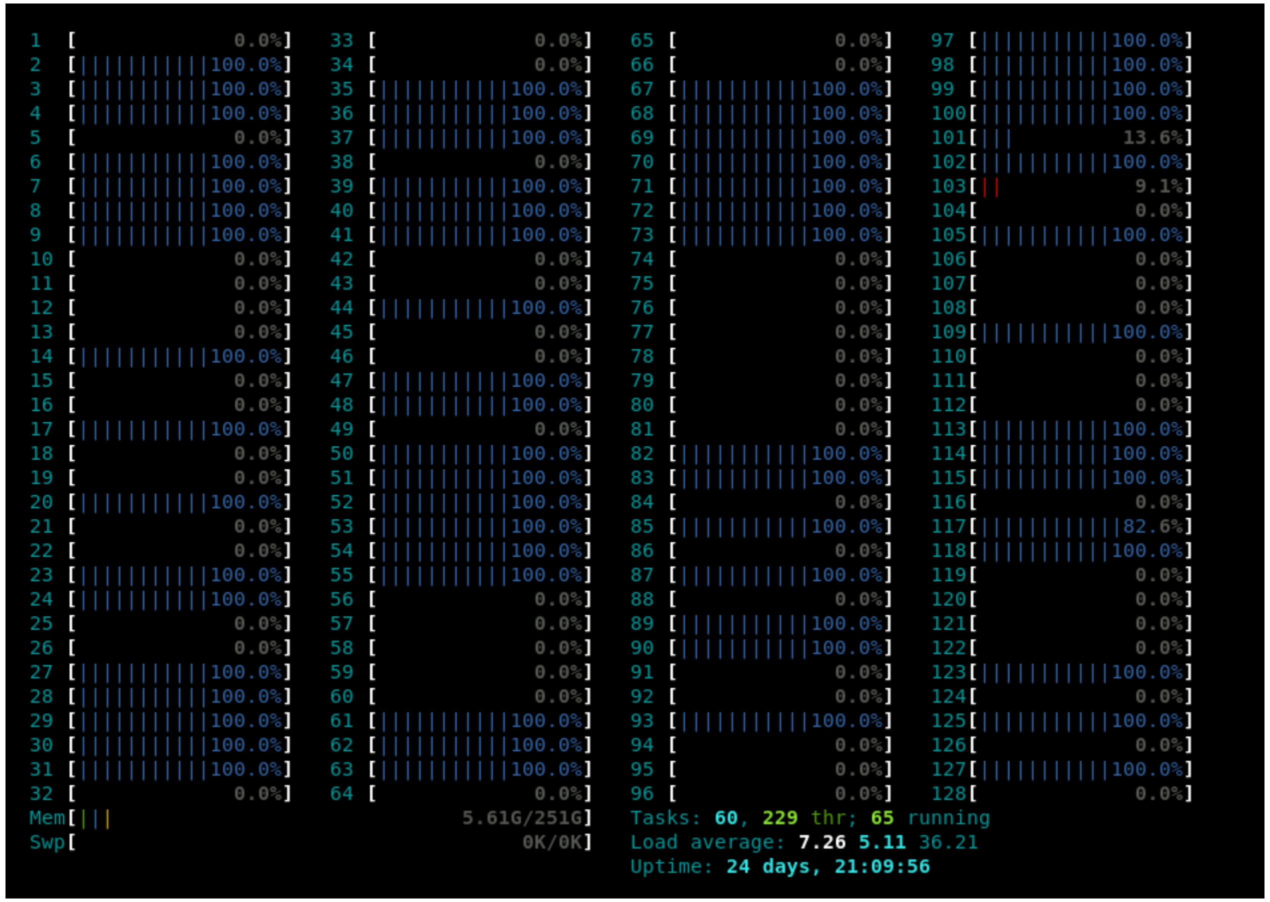
```
$ qsub -ADD-22-46 -qqprod -lselect=2:ncpus=128:mpiprocs=2:ompthreads=64 -I
$ mpirun -n 1 ./threaded
```

VSB TECHNICAL UNIVERSITY OF OSTRAVA | IT4INNOVATIONS NATIONAL SUPERCOMPUTING CENTER

PARALLEL RUN

How are threads pinned?

How will MPI be pinned?



```
1  [            0.0%]   33 [            0.0%]   65 [            0.0%]   97 [||||||||||100.0%]
2  [||||||||||100.0%]   34 [            0.0%]   66 [            0.0%]   98 [||||||||||100.0%]
3  [||||||||||100.0%]   35 [||||||||||100.0%]   67 [||||||||||100.0%]   99 [||||||||||100.0%]
4  [||||||||||100.0%]   36 [||||||||||100.0%]   68 [||||||||||100.0%]   100[||||||||||100.0%]
5  [            0.0%]   37 [||||||||||100.0%]   69 [||||||||||100.0%]   101[|||        13.6%]
6  [||||||||||100.0%]   38 [            0.0%]   70 [||||||||||100.0%]   102[||||||||||100.0%]
7  [||||||||||100.0%]   39 [||||||||||100.0%]   71 [||||||||||100.0%]   103[||          9.1%]
8  [||||||||||100.0%]   40 [||||||||||100.0%]   72 [||||||||||100.0%]   104[            0.0%]
9  [||||||||||100.0%]   41 [||||||||||100.0%]   73 [||||||||||100.0%]   105[||||||||||100.0%]
10 [            0.0%]   42 [            0.0%]   74 [            0.0%]   106[            0.0%]
11 [            0.0%]   43 [            0.0%]   75 [            0.0%]   107[            0.0%]
12 [            0.0%]   44 [||||||||||100.0%]   76 [            0.0%]   108[            0.0%]
13 [            0.0%]   45 [            0.0%]   77 [            0.0%]   109[||||||||||100.0%]
14 [||||||||||100.0%]   46 [            0.0%]   78 [            0.0%]   110[            0.0%]
15 [            0.0%]   47 [||||||||||100.0%]   79 [            0.0%]   111[            0.0%]
16 [            0.0%]   48 [||||||||||100.0%]   80 [            0.0%]   112[            0.0%]
17 [||||||||||100.0%]   49 [            0.0%]   81 [            0.0%]   113[||||||||||100.0%]
18 [            0.0%]   50 [||||||||||100.0%]   82 [||||||||||100.0%]   114[||||||||||100.0%]
19 [            0.0%]   51 [||||||||||100.0%]   83 [||||||||||100.0%]   115[||||||||||100.0%]
20 [||||||||||100.0%]   52 [||||||||||100.0%]   84 [            0.0%]   116[            0.0%]
21 [            0.0%]   53 [||||||||||100.0%]   85 [||||||||||100.0%]   117[|||||||||||82.6%]
22 [            0.0%]   54 [||||||||||100.0%]   86 [            0.0%]   118[||||||||||100.0%]
23 [||||||||||100.0%]   55 [||||||||||100.0%]   87 [||||||||||100.0%]   119[            0.0%]
24 [||||||||||100.0%]   56 [            0.0%]   88 [            0.0%]   120[            0.0%]
25 [            0.0%]   57 [            0.0%]   89 [||||||||||100.0%]   121[            0.0%]
26 [            0.0%]   58 [            0.0%]   90 [||||||||||100.0%]   122[            0.0%]
27 [||||||||||100.0%]   59 [            0.0%]   91 [            0.0%]   123[||||||||||100.0%]
28 [||||||||||100.0%]   60 [            0.0%]   92 [            0.0%]   124[            0.0%]
29 [||||||||||100.0%]   61 [||||||||||100.0%]   93 [||||||||||100.0%]   125[||||||||||100.0%]
30 [||||||||||100.0%]   62 [||||||||||100.0%]   94 [            0.0%]   126[            0.0%]
31 [||||||||||100.0%]   63 [||||||||||100.0%]   95 [            0.0%]   127[||||||||||100.0%]
32 [            0.0%]   64 [            0.0%]   96 [            0.0%]   128[            0.0%]
Mem[|||                           5.61G/251G]   Tasks: 60, 229 thr; 65 running
Swp[                                 0K/0K]   Load average: 7.26 5.11 36.21
                                              Uptime: 24 days, 21:09:56
```

Unfortunately:

- pinning significantly influence performance
- pinning is **highly non-portable**
  - different settings for OpenMPI, Intel
  - dependent on a particular system

```
$ qsub -ADD-22-46 -qqprod -lselect=2:ncpus=128:mpiprocs=2:ompthreads=64 -I
$ mpirun -n 1 ./threaded
```

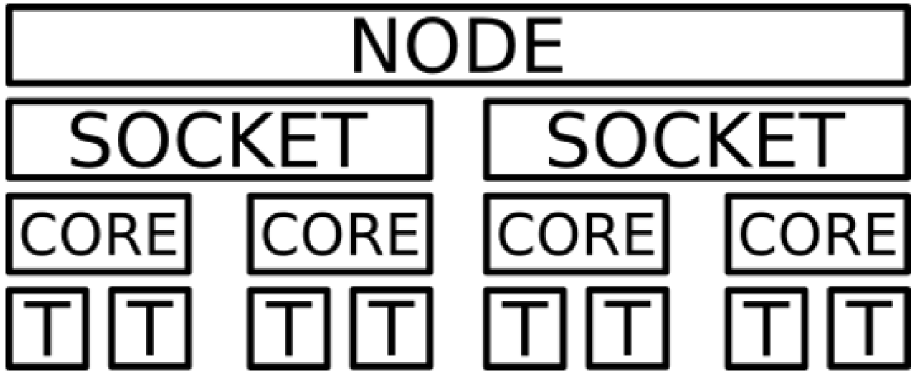
PARALLEL RUN

Environment variables

- OMP_NUM_THREADS
- OMP_PLACES=<threads, cores, sockets>
- OMP_PROC_BIND=<true, false, master, close, spread>

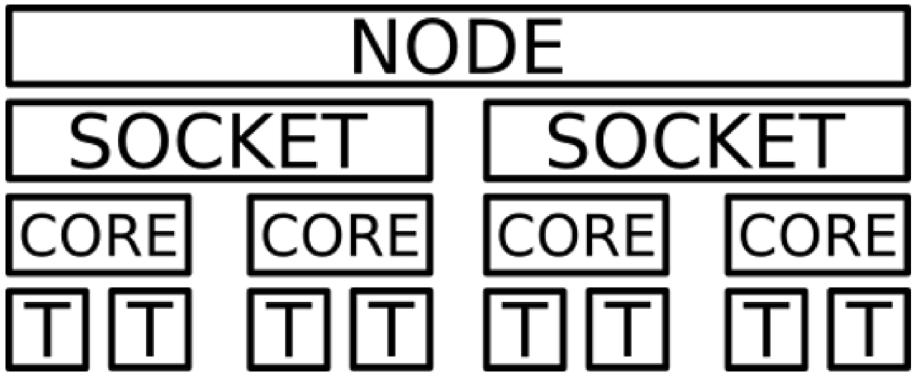
- Intel-MPI
  - KMP_AFFINITY
  - I_MPI_PIN_DOMAIN
- OpenMPI
  - --bind-to <hwthread, core, socket, numa, ...>
  - --map-by <hwthread, core, socket, numa, ...>
  - --report-bindings

PARALLEL RUN – INTEL MPI



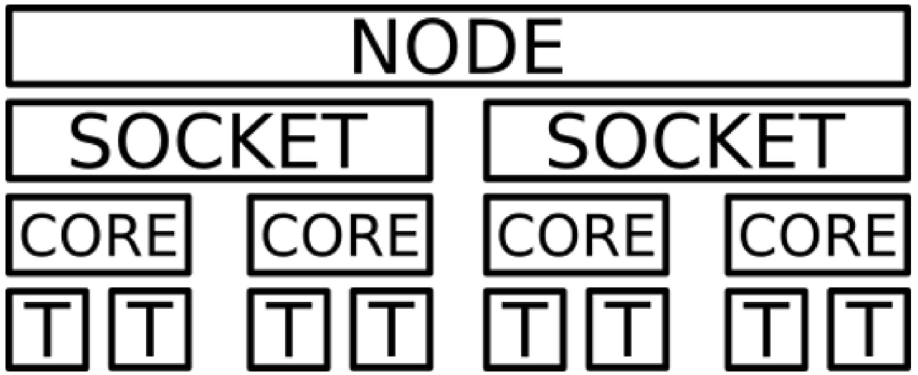
OMP_PROC_BIND=close 0 1 2 3 4 5 6 7

PARALLEL RUN – INTEL MPI



```
OMP_PROC_BIND=close                              0   1   2   3   4   5   6   7
OMP_NUM_THREADS=2 OMP_PROC_BIND=spread           0               1
```

PARALLEL RUN – INTEL MPI



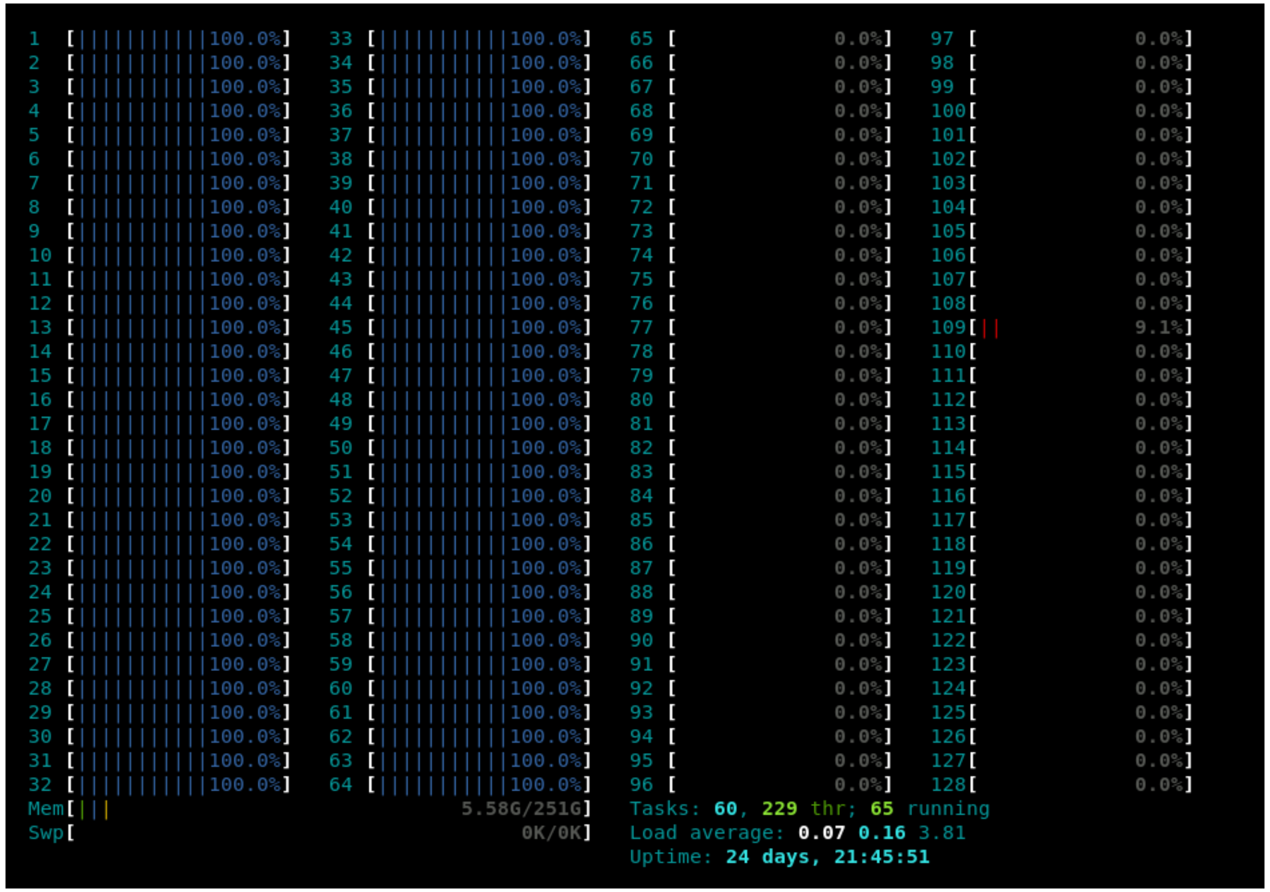
OMP_PROC_BIND=close	0	1	2	3	4	5	6	7
OMP_NUM_THREADS=2 OMP_PROC_BIND=spread	0				1			
OMP_NUM_THREADS=4 OMP_PROC_BIND=spread	0		1		2		3	

PARALLEL RUN – INTEL MPI

```
$ qsub -ADD-22-46 -qqprod -lselect=2:ncpus=128:mpiprocs=2:ompthreads=64 -I

$ export OMP_PROC_BIND=close
$ mpirun -n 1 ./threaded
```

my application is 2x faster!

```
$ ssh cnXXX
$ htop -d2
```



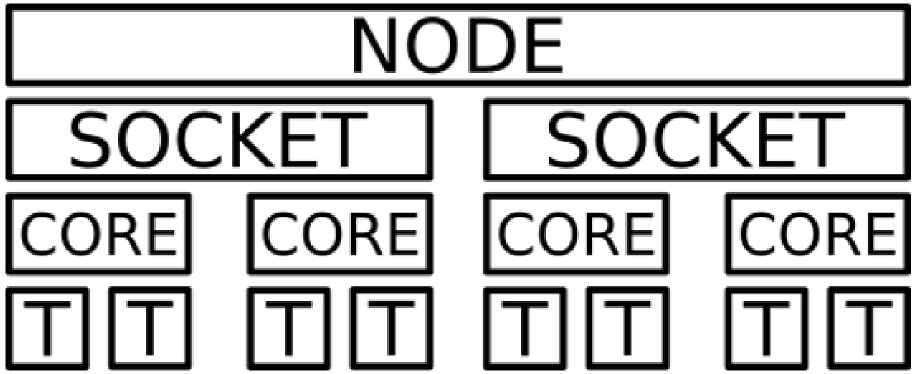
```
1  [||||||||||100.0%]   33 [||||||||||100.0%]   65 [          0.0%]   97 [          0.0%]
2  [||||||||||100.0%]   34 [||||||||||100.0%]   66 [          0.0%]   98 [          0.0%]
3  [||||||||||100.0%]   35 [||||||||||100.0%]   67 [          0.0%]   99 [          0.0%]
4  [||||||||||100.0%]   36 [||||||||||100.0%]   68 [          0.0%]   100[          0.0%]
5  [||||||||||100.0%]   37 [||||||||||100.0%]   69 [          0.0%]   101[          0.0%]
6  [||||||||||100.0%]   38 [||||||||||100.0%]   70 [          0.0%]   102[          0.0%]
7  [||||||||||100.0%]   39 [||||||||||100.0%]   71 [          0.0%]   103[          0.0%]
8  [||||||||||100.0%]   40 [||||||||||100.0%]   72 [          0.0%]   104[          0.0%]
9  [||||||||||100.0%]   41 [||||||||||100.0%]   73 [          0.0%]   105[          0.0%]
10 [||||||||||100.0%]   42 [||||||||||100.0%]   74 [          0.0%]   106[          0.0%]
11 [||||||||||100.0%]   43 [||||||||||100.0%]   75 [          0.0%]   107[          0.0%]
12 [||||||||||100.0%]   44 [||||||||||100.0%]   76 [          0.0%]   108[          0.0%]
13 [||||||||||100.0%]   45 [||||||||||100.0%]   77 [          0.0%]   109[||        9.1%]
14 [||||||||||100.0%]   46 [||||||||||100.0%]   78 [          0.0%]   110[          0.0%]
15 [||||||||||100.0%]   47 [||||||||||100.0%]   79 [          0.0%]   111[          0.0%]
16 [||||||||||100.0%]   48 [||||||||||100.0%]   80 [          0.0%]   112[          0.0%]
17 [||||||||||100.0%]   49 [||||||||||100.0%]   81 [          0.0%]   113[          0.0%]
18 [||||||||||100.0%]   50 [||||||||||100.0%]   82 [          0.0%]   114[          0.0%]
19 [||||||||||100.0%]   51 [||||||||||100.0%]   83 [          0.0%]   115[          0.0%]
20 [||||||||||100.0%]   52 [||||||||||100.0%]   84 [          0.0%]   116[          0.0%]
21 [||||||||||100.0%]   53 [||||||||||100.0%]   85 [          0.0%]   117[          0.0%]
22 [||||||||||100.0%]   54 [||||||||||100.0%]   86 [          0.0%]   118[          0.0%]
23 [||||||||||100.0%]   55 [||||||||||100.0%]   87 [          0.0%]   119[          0.0%]
24 [||||||||||100.0%]   56 [||||||||||100.0%]   88 [          0.0%]   120[          0.0%]
25 [||||||||||100.0%]   57 [||||||||||100.0%]   89 [          0.0%]   121[          0.0%]
26 [||||||||||100.0%]   58 [||||||||||100.0%]   90 [          0.0%]   122[          0.0%]
27 [||||||||||100.0%]   59 [||||||||||100.0%]   91 [          0.0%]   123[          0.0%]
28 [||||||||||100.0%]   60 [||||||||||100.0%]   92 [          0.0%]   124[          0.0%]
29 [||||||||||100.0%]   61 [||||||||||100.0%]   93 [          0.0%]   125[          0.0%]
30 [||||||||||100.0%]   62 [||||||||||100.0%]   94 [          0.0%]   126[          0.0%]
31 [||||||||||100.0%]   63 [||||||||||100.0%]   95 [          0.0%]   127[          0.0%]
32 [||||||||||100.0%]   64 [||||||||||100.0%]   96 [          0.0%]   128[          0.0%]
Mem[|||                          5.58G/251G]   Tasks: 60, 229 thr; 65 running
Swp[                                 0K/0K]   Load average: 0.07 0.16 3.81
                                              Uptime: 24 days, 21:45:51
```

PARALLEL RUN – INTEL MPI

- KMP_AFFINITY=[<modifier>,...]<type>[,<permute>][,<offset>]
  - modifier:
    - verbose, warnings, respect
    - granularity= fine, **thread**, core, tile, die, node, group, and socket
  - type:
    - balanced, **compact**, disabled, explicit, none, scatter
  - permute
    - 0 – thread, 1 – core, 2 – socket
    - positive number (default 0)
  - offset
    - position where the first thread is assigned
    - positive number (default 0)

https://www.intel.com/content/www/us/en/develop/documentation/mpi-developer-reference-linux/top/environment-variable-reference/process-pinning/environment-variables-for-process-pinning.html

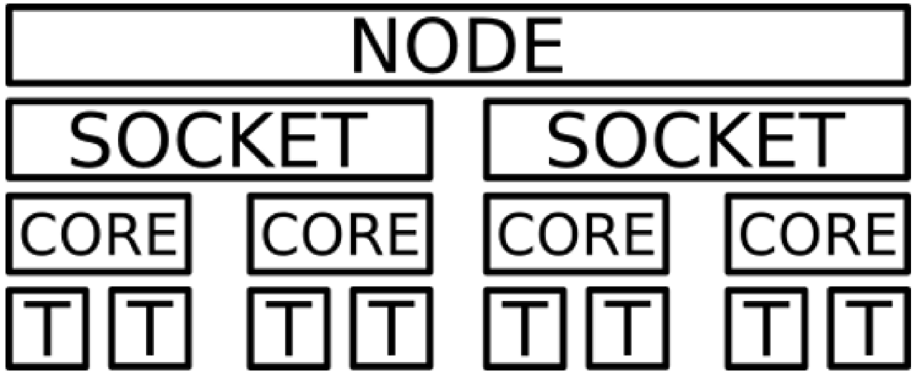
PARALLEL RUN – INTEL MPI



KMP_AFFINITY=granularity=thread,compact

0 1 2 3 4 5 6 7

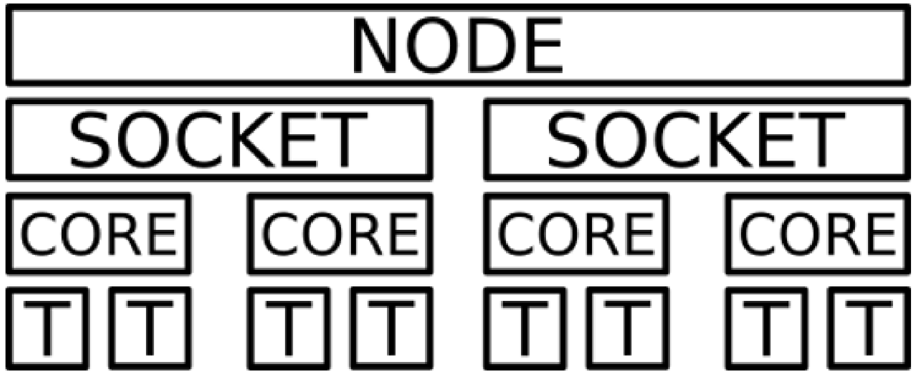
PARALLEL RUN – INTEL MPI



KMP_AFFINITY=granularity=thread,compact	0 1 2 3 4 5 6 7
KMP_AFFINITY=granularity=thread,scatter	0

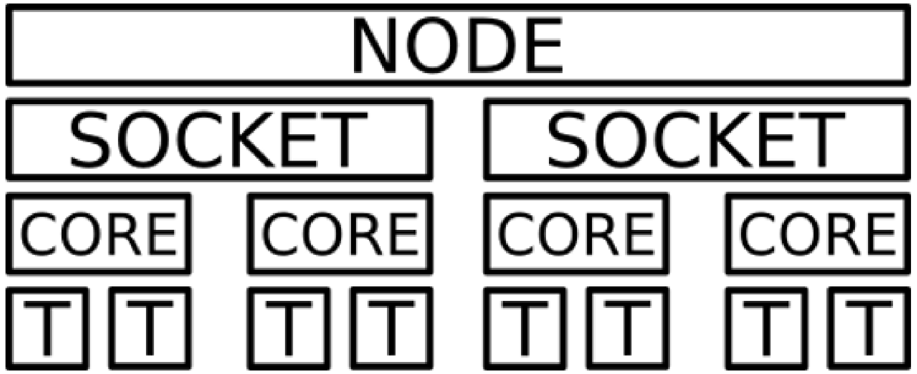
VSB TECHNICAL UNIVERSITY OF OSTRAVA | IT4INNOVATIONS NATIONAL SUPERCOMPUTING CENTER

PARALLEL RUN – INTEL MPI



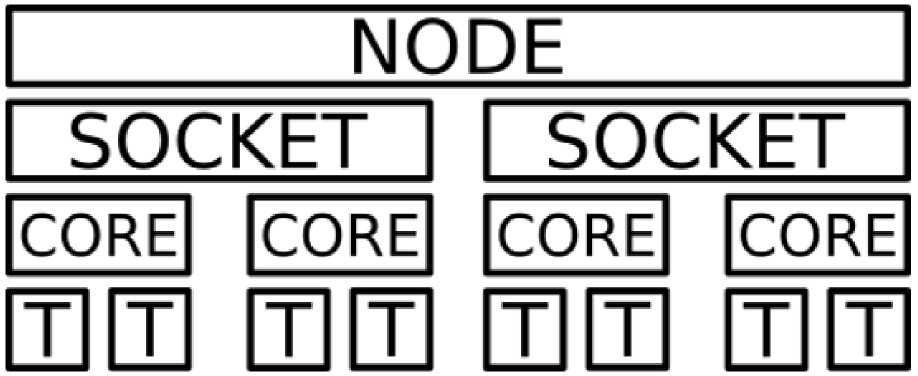
KMP_AFFINITY=granularity=thread,compact	0	1	2	3	4	5	6	7
KMP_AFFINITY=granularity=thread,scatter	0				1			

PARALLEL RUN – INTEL MPI



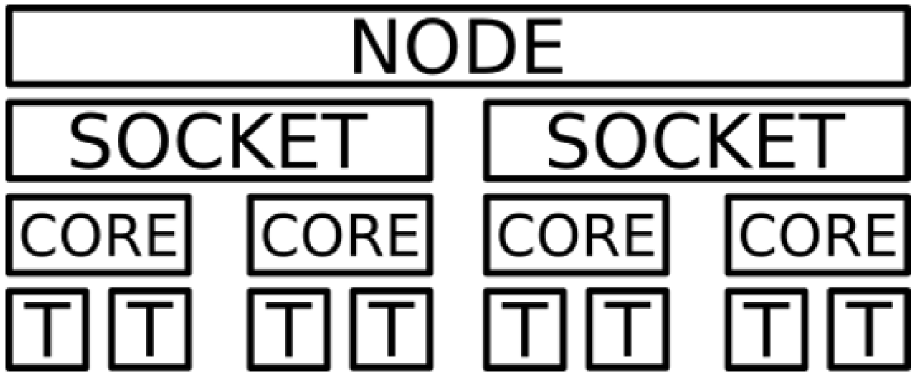
KMP_AFFINITY=granularity=thread,compact	0	1	2	3	4	5	6	7
KMP_AFFINITY=granularity=thread,scatter	0		2		1		3	

PARALLEL RUN – INTEL MPI



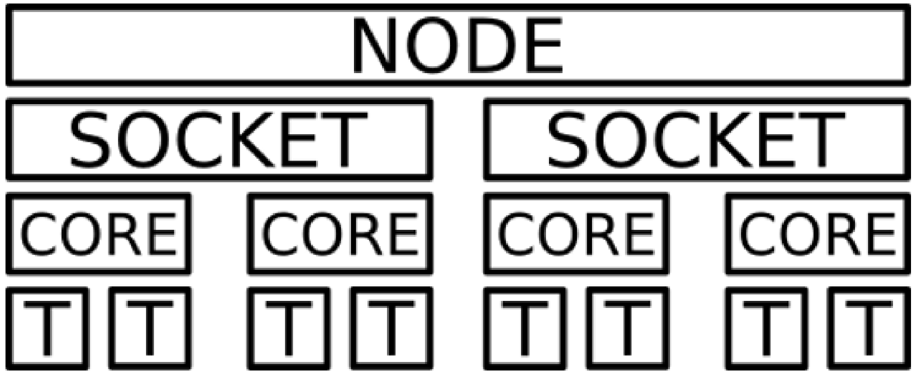
KMP_AFFINITY=granularity=thread,compact	0	1	2	3	4	5	6	7
KMP_AFFINITY=granularity=thread,scatter	0	4	2	6	1	5	3	7
KMP_AFFINITY=granularity=thread,compact,0,5								

PARALLEL RUN – INTEL MPI



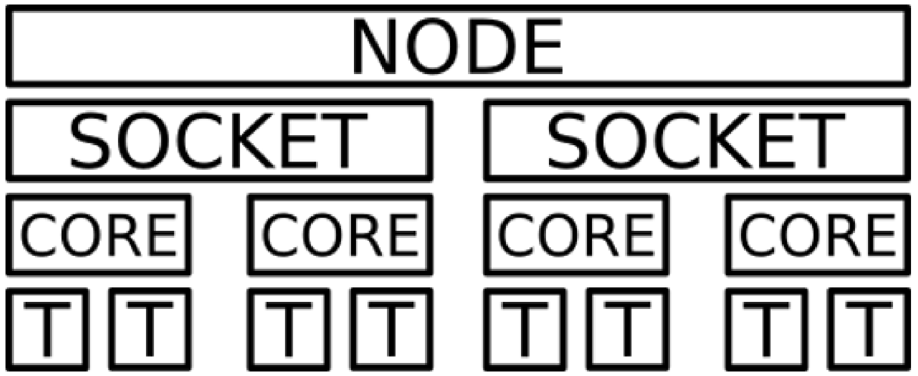
KMP_AFFINITY=granularity=thread,compact	0	1	2	3	4	5	6	7
KMP_AFFINITY=granularity=thread,scatter	0	4	2	6	1	5	3	7
KMP_AFFINITY=granularity=thread,compact,0,5						0		

PARALLEL RUN – INTEL MPI



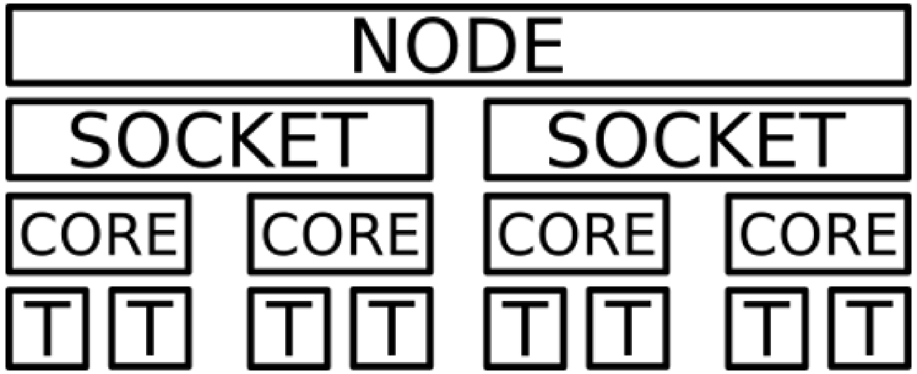
KMP_AFFINITY=granularity=thread,compact	0	1	2	3	4	5	6	7
KMP_AFFINITY=granularity=thread,scatter	0	4	2	6	1	5	3	7
KMP_AFFINITY=granularity=thread,compact,0,5	3	4	5	6	7	0	1	2

PARALLEL RUN – INTEL MPI



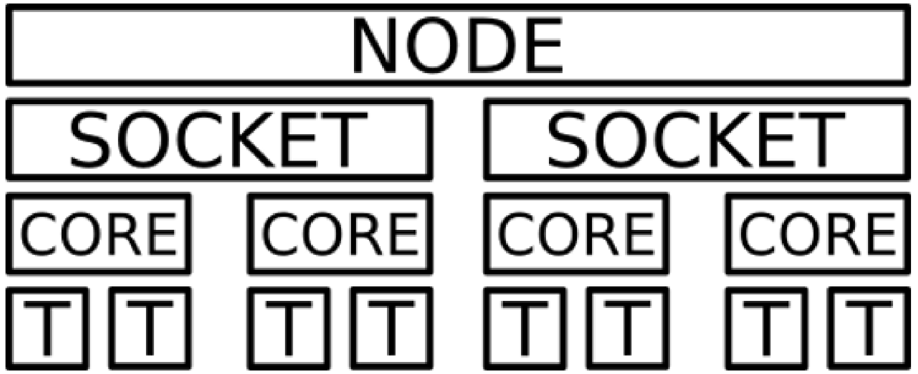
KMP_AFFINITY=granularity=thread,compact	0	1	2	3	4	5	6	7
KMP_AFFINITY=granularity=thread,scatter	0	4	2	6	1	5	3	7
KMP_AFFINITY=granularity=thread,compact,0,5	3	4	5	6	7	0	1	2
KMP_AFFINITY=granularity=thread,compact,1,0								

PARALLEL RUN – INTEL MPI



KMP_AFFINITY=granularity=thread,compact	0	1	2	3	4	5	6	7
KMP_AFFINITY=granularity=thread,scatter	0	4	2	6	1	5	3	7
KMP_AFFINITY=granularity=thread,compact,0,5	3	4	5	6	7	0	1	2
KMP_AFFINITY=granularity=thread,compact,1,0	0		1		2		3	

PARALLEL RUN – INTEL MPI

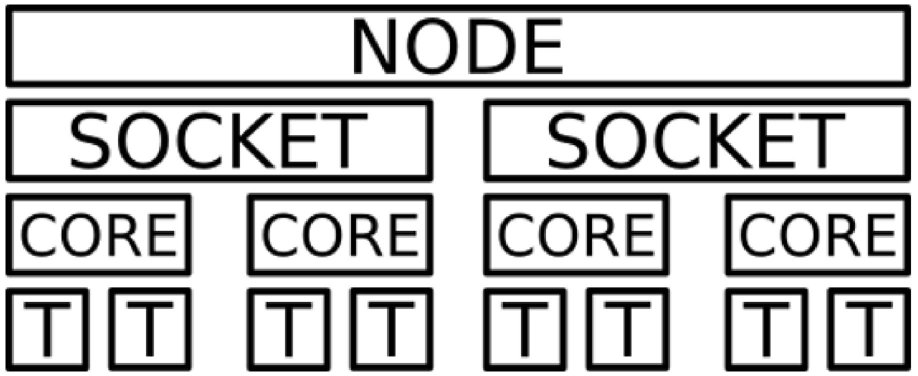


KMP_AFFINITY=granularity=thread,compact	0	1	2	3	4	5	6	7
KMP_AFFINITY=granularity=thread,scatter	0	4	2	6	1	5	3	7
KMP_AFFINITY=granularity=thread,compact,0,5	3	4	5	6	7	0	1	2
KMP_AFFINITY=granularity=thread,compact,1,0	0	4	1	5	2	6	3	7

PARALLEL RUN – INTEL MPI

- I_MPI_PIN_DOMAIN=[shape]
    - <size>[:<layout>]
        - number of logical processors in each domain with a layout (platform, compact, scatter)
    - core
    - socket
    - numa
    - cache

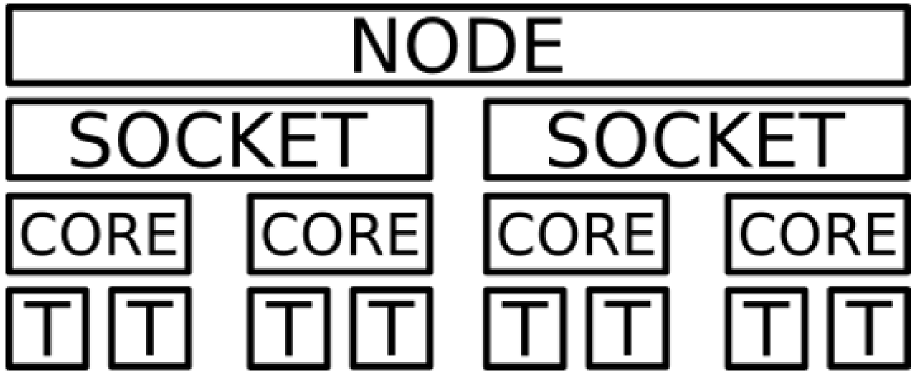
PARALLEL RUN – INTEL MPI



I_MPI_PIN_DOMAIN=4

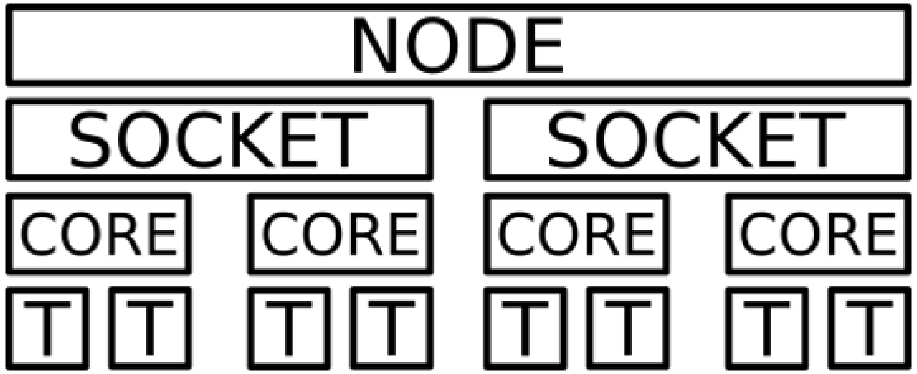
```
0   0   0   0   1   1   1   1
```

PARALLEL RUN – INTEL MPI



I_MPI_PIN_DOMAIN=4	0	0	0	0	1	1	1	1
I_MPI_PIN_DOMAIN=2	0	0	1	1	2	2	3	3

PARALLEL RUN – INTEL MPI



I_MPI_PIN_DOMAIN=4	0	0	0	0	1	1	1	1
I_MPI_PIN_DOMAIN=2	0	0	1	1	2	2	3	3

I_MPI_PIN_DOMAIN=$OMP_NUM_THREADS

I_MPI_PIN_DOMAIN=socket

I_MPI_PIN_DOMAIN=cache3

I_MPI_PIN_RESPECT_HCA=0   pinning does not respect host channel adapter

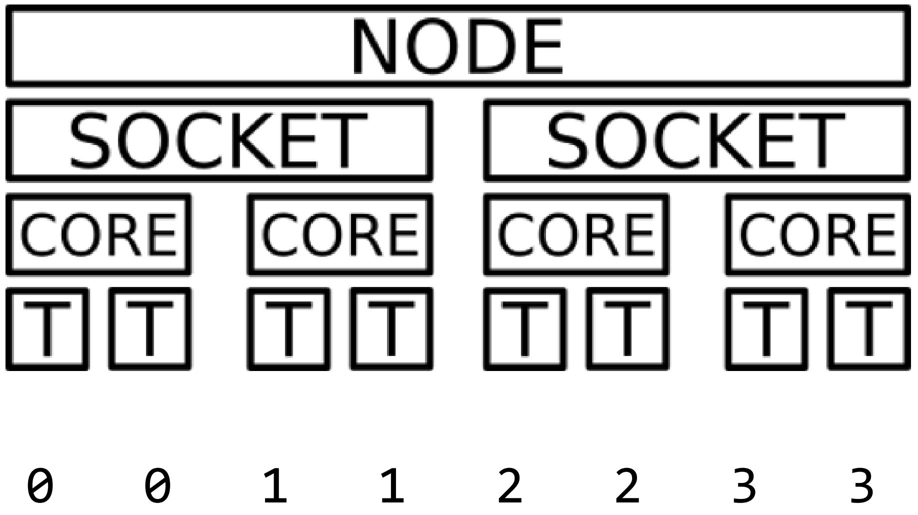
PARALLEL RUN – OPEN MPI

- --bind-to <hwthread, **core**, l3cache, numa, socket, ppr, …>
  - bind to the processors associated with hardware component
- --map-by <hwthread, core, l3cache, numa, **socket**, …>
  - map across the specified hardware component
- --report-bindings

https://www.open-mpi.org/doc/v3.0/man1/mpirun.1.php

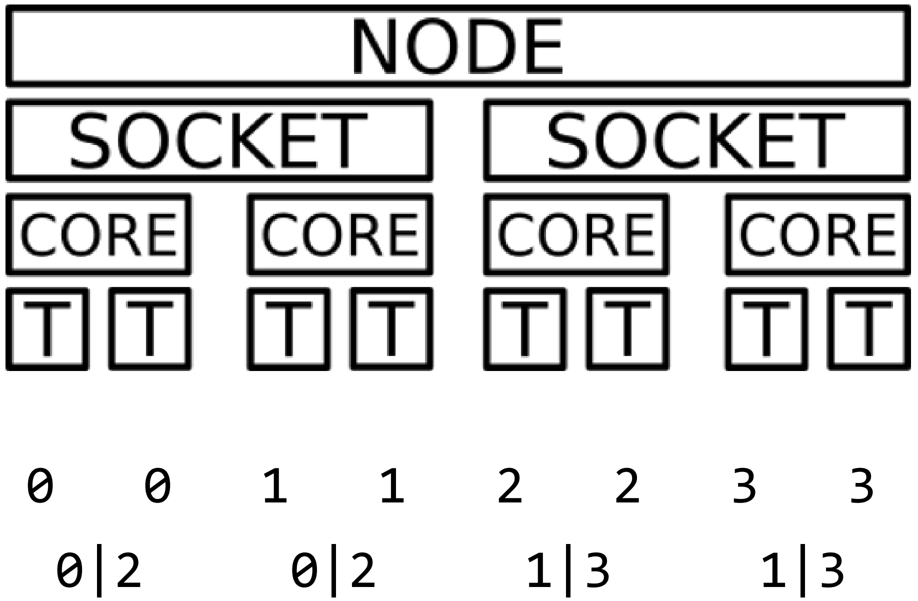
PARALLEL RUN – OPEN MPI

```
export OMP_PROC_BIND=close
export OMP_NUM_THREADS=2
mpirun -n 4 --map-by core --bind-to core ./app
```

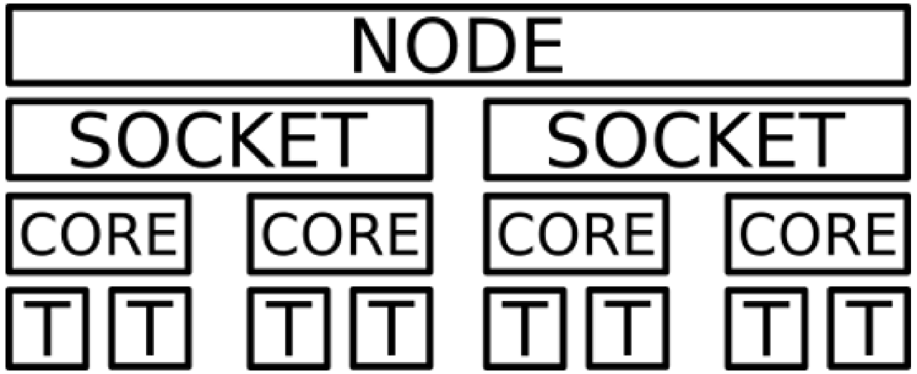


PARALLEL RUN – OPEN MPI

```
export OMP_PROC_BIND=close
export OMP_NUM_THREADS=2
mpirun -n 4 --map-by core --bind-to core ./app
mpirun -n 4 --map-by socket --bind-to socket ./app
```

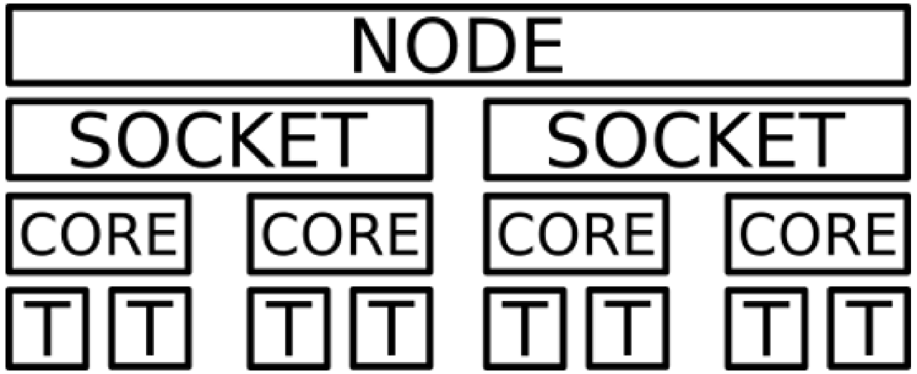


PARALLEL RUN – OPEN MPI



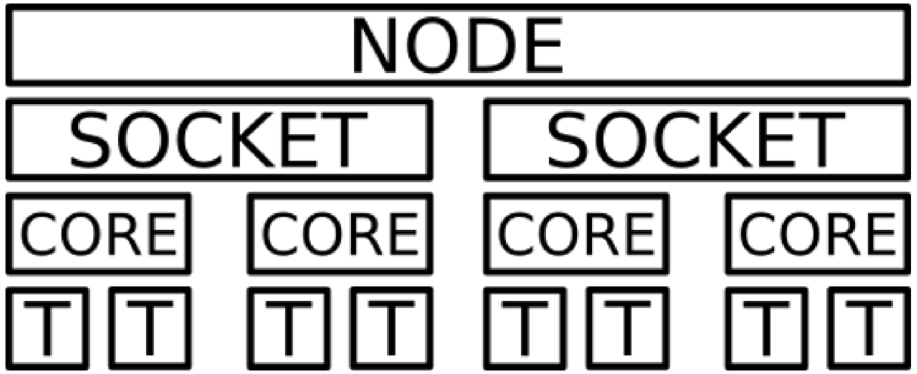
```
export OMP_PROC_BIND=close
export OMP_NUM_THREADS=2
mpirun -n 4 --map-by core --bind-to core ./app            0   0   1   1   2   2   3   3
mpirun -n 4 --map-by socket --bind-to socket ./app         0|2     0|2     1|3     1|3
mpirun -n 4 --map-by socket --bind-to core ./app           0|2 -  -   -    1|3 -  -   -
```

PARALLEL RUN – OPEN MPI



```
export OMP_PROC_BIND=close
export OMP_NUM_THREADS=2
mpirun -n 4 --map-by core --bind-to core ./app            0   0   1   1   2   2   3   3
mpirun -n 4 --map-by socket --bind-to socket ./app         0|2     0|2     1|3     1|3
mpirun -n 4 --map-by socket --bind-to core ./app           0|2 -  -   -    1|3 -  -   -
mpirun -n 4 --map-by thread --bind-to socket ./app        0|1|2|3 0|1|2|3  ---     ---
```

PARALLEL RUN – OPEN MPI

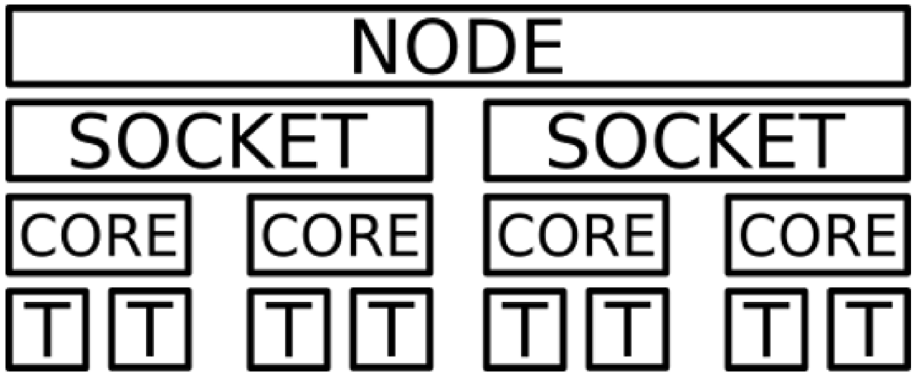


export OMP_PROC_BIND=close

export OMP_NUM_THREADS=2

mpirun -n 4 --map-by core --bind-to core ./app	`0   0   1   1   2   2   3   3`
mpirun -n 4 --map-by socket --bind-to socket ./app	`0|2     0|2     1|3     1|3`
mpirun -n 4 --map-by socket --bind-to core ./app	`0|2 -   -   -    1|3 -   -   -`
mpirun -n 4 --map-by thread --bind-to socket ./app	`0|1|2|3 0|1|2|3  ---      ---`
mpirun -n 4 --map-by numa --bind-to numa ./app	`0|2     0|2     1|3     1|3`

PARALLEL RUN – OPEN MPI



```
export OMP_PROC_BIND=close
export OMP_NUM_THREADS=2
mpirun -n 4 --map-by core --bind-to core ./app             0   0   1   1   2   2   3   3
mpirun -n 4 --map-by socket --bind-to socket ./app          0|2     0|2     1|3     1|3
mpirun -n 4 --map-by socket --bind-to core ./app            0|2 -   -   -   1|3 -   -   -
mpirun -n 4 --map-by thread --bind-to socket ./app         0|1|2|3 0|1|2|3  ---     ---
mpirun -n 4 --map-by numa --bind-to numa ./app              0|2     0|2     1|3     1|3
```

OpenMPI defaults
--bin-to core   (when the number of processes is <= 2)
--bind-to socket (when the number of processes is   > 2)

PARALLEL RUN

What is the optimal setting?

PARALLEL RUN

What is the optimal setting?

- hardware configuration
  - number of NUMA domains
  - caches, memory channels,...

- application features
  - OpenMP only
  - pure MPI
  - hybrid parallelization

PARALLEL RUN

Memory bound application

- number of MPI processes / thread equal to memory channels
- correct pinning to NUMA domains (sockets, chiplets)

Compute bound application

- as many MPI processes / threads as possible

Your application?

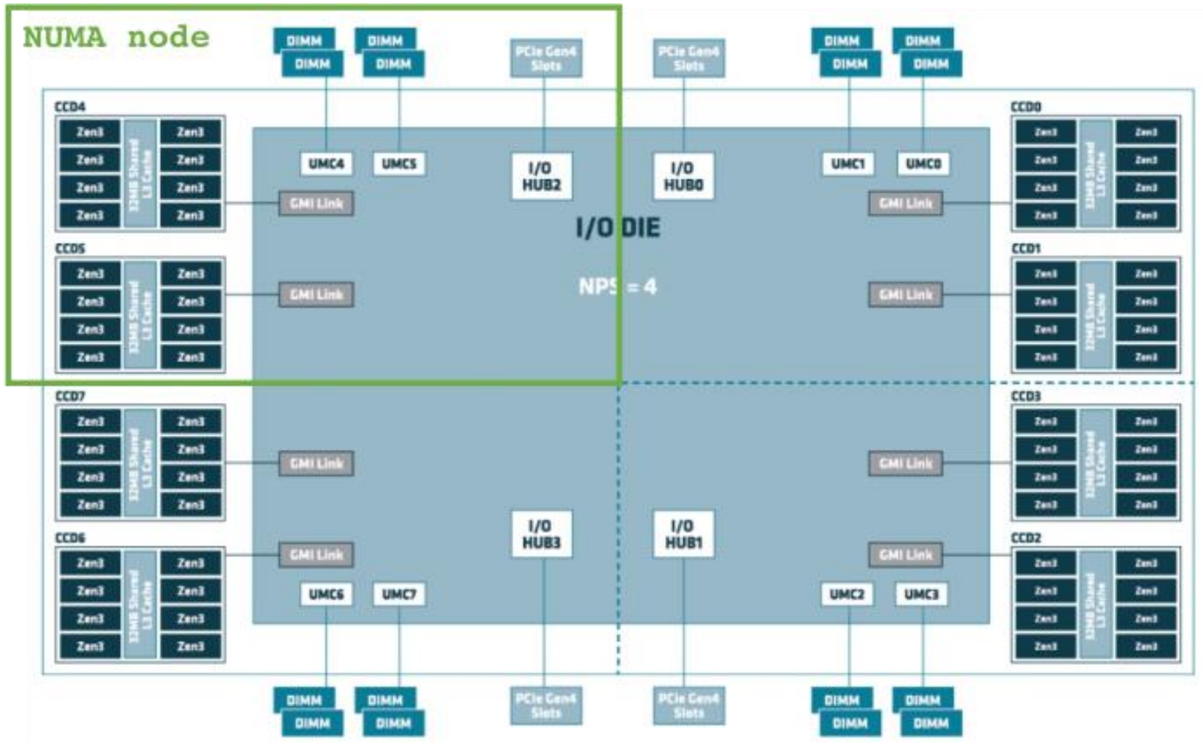
- one MPI process per NUMA domain
- number of cores in NUMA domain

KAROLINA

Node architecture

```
numactl -H
| node 0 cpus:   0 -  15
| node 1 cpus:  16 -  31
| node 2 cpus:  32 -  47
| node 3 cpus:  48 -  63
| node 4 cpus:  64 -  79
| node 5 cpus:  80 -  95
| node 6 cpus:  96 - 111
| node 7 cpus: 112 - 127
| node 0-7 size: 128GB
```

	0	1	2	3	4	5	6	7
0	10	12	12	12	32	32	32	32
1	12	10	12	12	32	32	32	32
2	12	12	10	12	32	32	32	32
3	12	12	12	10	32	32	32	32
4	32	32	32	32	10	12	12	12
5	32	32	32	32	12	10	12	12
6	32	32	32	32	12	12	10	12
7	32	32	32	32	12	12	12	10



PARALLEL RUN – INTEL MPI

```
$ qsub -ADD-22-46 -qqprod -lselect=1:ncpus=128:mpiprocs=8:ompthreads=16 -I
```

src/sequential.cpp

```
double *vals = new double[rows * cols];


for (int r = 0; r < rows; ++r) {
    for (int c = 0; c < cols; ++c) {
        vals[r * cols + c] = 0;
    }
}
```

src/threaded.cpp

```
double *vals = new double[rows * cols];

#pragma omp parallel for collapse(2)
for (int r = 0; r < rows; ++r) {
    for (int c = 0; c < cols; ++c) {
        vals[r * cols + c] = 0;
    }
}
```

PARALLEL RUN – INTEL MPI

```
$ qsub -ADD-22-46 -qqprod -lselect=1:ncpus=128:mpiprocs=8:ompthreads=16 -I

$ OMP_NUM_THREADS=64 mpirun -n 2 ./sequential  -> 55s

$ OMP_NUM_THREADS=64 mpirun -n 2 --bind-to socket ./sequential  -> 88s

$ OMP_NUM_THREADS=64 mpirun -n 2 --bind-to socket -map-by socket ./sequential  -> 3.6s

$ OMP_NUM_THREADS=64 OMP_PROC_BIND=close mpirun -n 2 --bind-to socket --map-by socket ./sequential  -> 3.6s


$ OMP_NUM_THREADS=64 mpirun -n 2 ./threaded  -> 55s

$ OMP_NUM_THREADS=64 mpirun -n 2 --bind-to socket ./threaded  -> 86s

$ OMP_NUM_THREADS=64 mpirun -n 2 --bind-to socket --map-by socket ./threaded  -> 6.8s

$ OMP_NUM_THREADS=64 OMP_PROC_BIND=close mpirun -n 2 --bind-to socket --map-by socket ./threaded  -> 1.2s


$ OMP_NUM_THREADS=16 OMP_PROC_BIND=close mpirun -n 8 --bind-to numa --map-by numa ./sequential  -> 0.9s
$ OMP_NUM_THREADS=16 OMP_PROC_BIND=close mpirun -n 8 --bind-to numa --map-by numa ./threaded    -> 0.9s
```

Ondřej Meca
ondrej.meca@vsb.cz

IT4Innovations National Supercomputing Center
VSB – Technical University of Ostrava
Studentská 6231/1B
708 00 Ostrava-Poruba, Czech Republic
www.it4i.cz

VSB TECHNICAL UNIVERSITY OF OSTRAVA | IT4INNOVATIONS NATIONAL SUPERCOMPUTING CENTER

EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research, Development and Education

MINISTRY OF EDUCATION, YOUTH AND SPORTS