TACC
TEXAS ADVANCED COMPUTING CENTER
WWW.TACC.UTEXAS.EDU

TEXAS
The University of Texas at Austin

IXPUG-ISC16 Workshop

OpenMP Affinity.. ..on KNL

PRESENTED BY:

Kent Milfeld

milfeld@tacc.utexas.edu

Affinity

Why do we need it?

- Some Apps run better with fewer threads than HW-threads.
  Execution with only 1, 2, or 3 threads per core.
  Execution with only a few threads per tile.
  Execution with fewer cores per quadrant (e.g. avoid #6)
  Allow threads to float on cores.

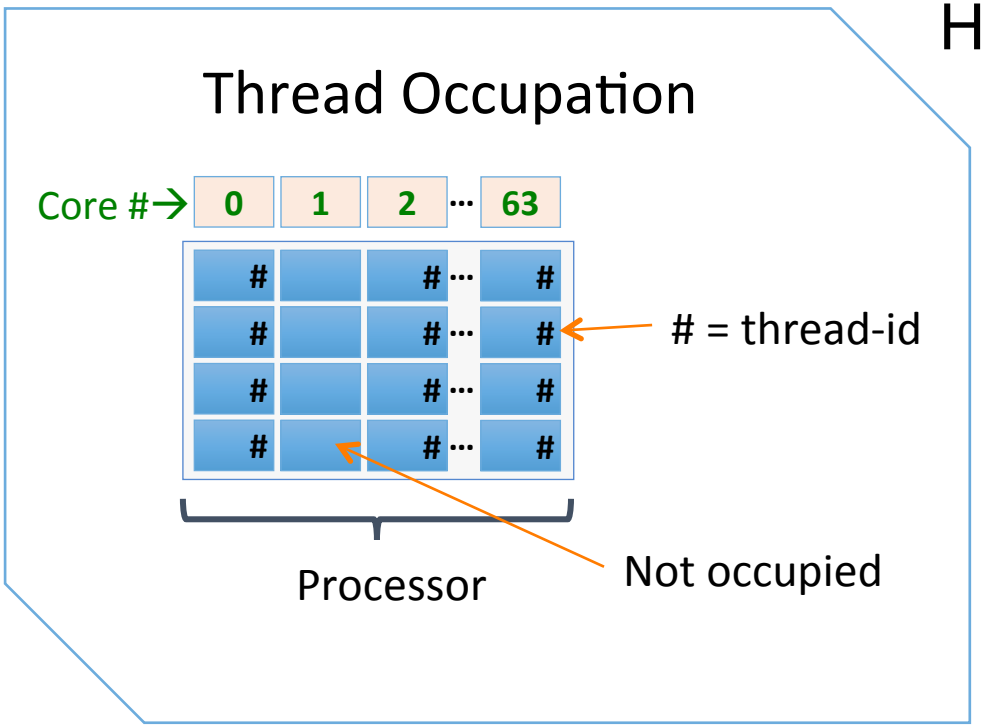
What are the OpenMP tools?

- distribution policy (close/spread …)
- syntax for expressing HW-threads (places) to run on

Affinity (thread-id assignment)

In a parallel region threads are assigned to HW-threads.

We will now show occupation: assigned thread-ids on the HW-thread “grid”.



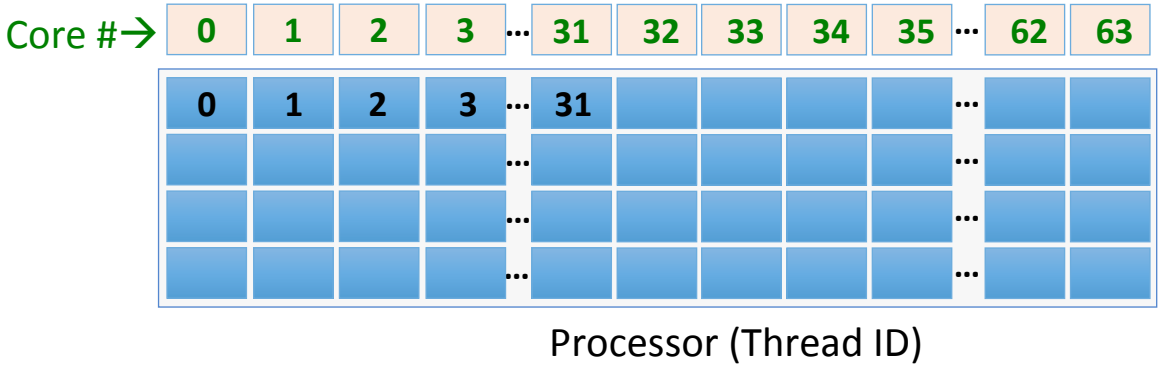
Layout is for a 64-core system

Affinity (distribution)

Distributions (fewer threads than the total 64 cores):

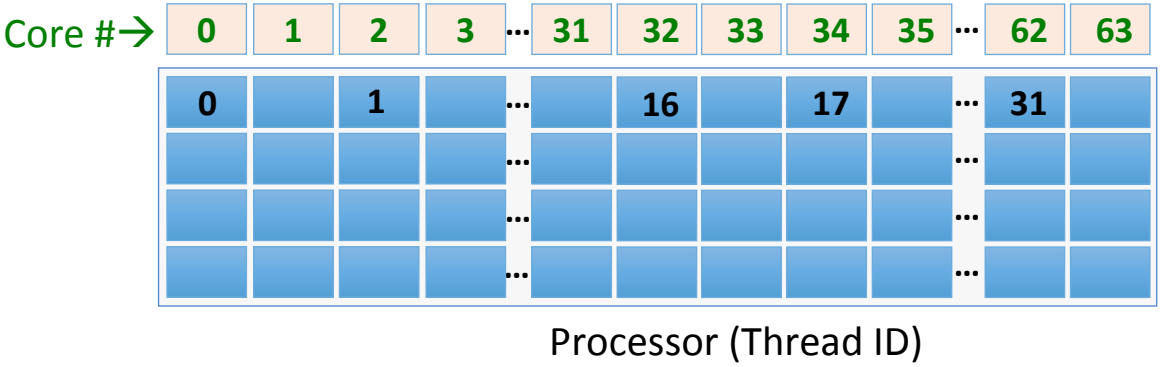
close (keeps threads together)

```
OMP_NUM_THREADS=32
OMP_PROC_BIND=close
```



spread (spreads threads out across cores)

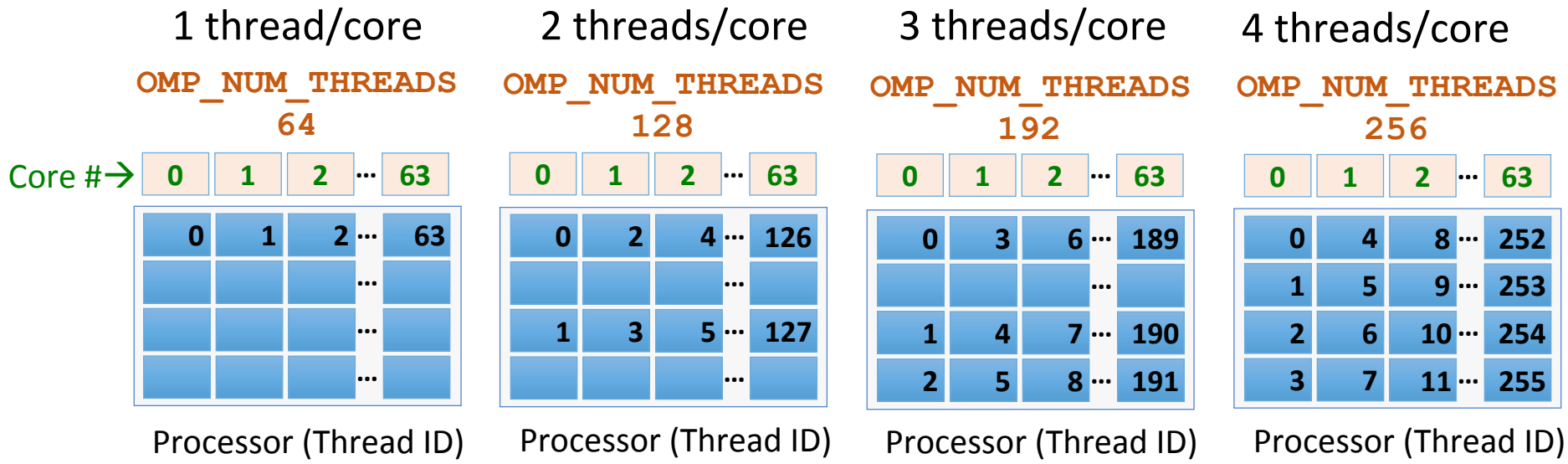
```
OMP_NUM_THREADS=32
OMP_PROC_BIND=spread
```



Affinity (distribution)

Distributions (greater number of threads than the total 64 cores):
Occupying 1, 2, 3 and 4 OpenMP threads per core:

```
export OMP_PROC_BIND=spread
```

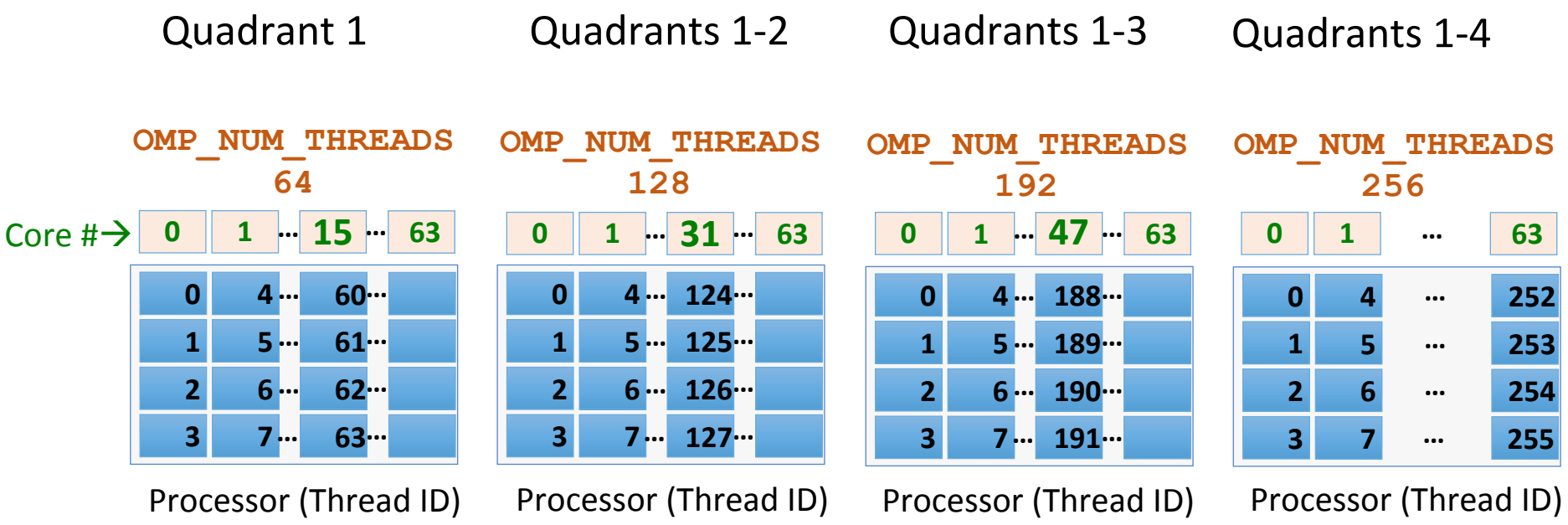


Sequential threads are on the same core.
Try this affinity setting when increasing the threads per core.

Affinity (distribution)

Distributions (greater number of threads than the total 64 cores):
Occupying 1, 2, 3 and 4 of the quadrants:

```
export OMP_PROC_BIND=close
```



Affinity (places)

- A PLACE is a list of HW-threads.

  Comma separated list e.g. {0},{1},{2},{3},{4}

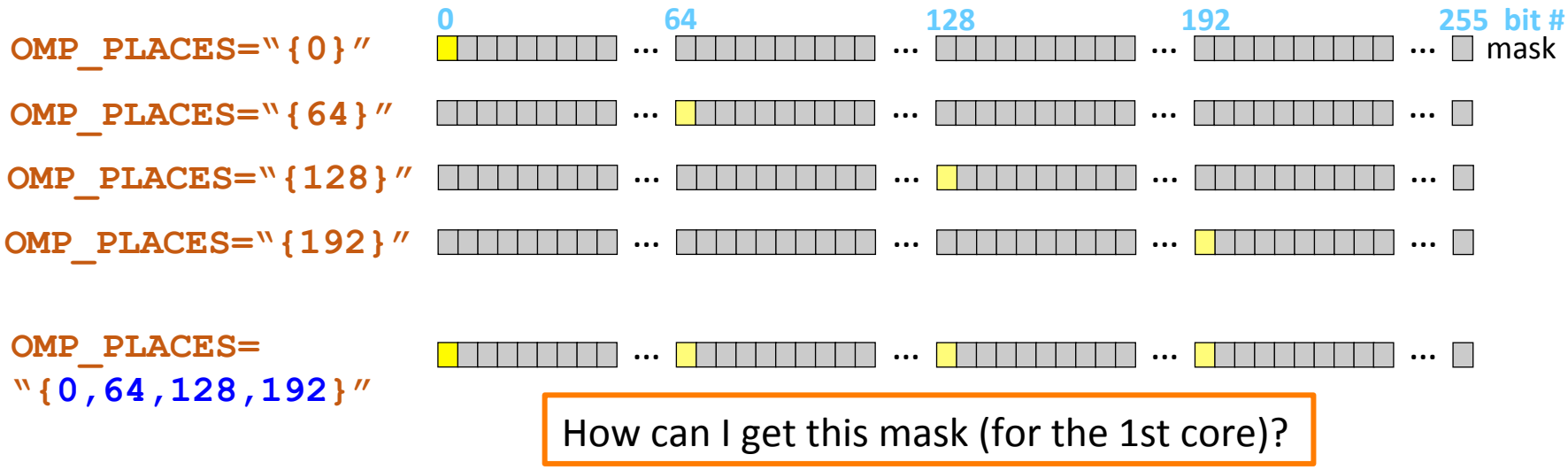
- Basic place syntax: *{ lower-bound } : length : stride*

  { } : length e.g. {0}:5 = {0},{1},{2},{3},{4}
  { } : length : stride e.g. {0}:3:2 = {0},{2},{4}

```
e.g. export OMP_NUM_THREADS=128  OMP_PLACES="{0}:128"
```

Affinity (HW-thread maps, floating threads)

A 256-bit mask specifies where a thread may execute.



How can I get this mask (for the 1st core)?

A list within a place is a mask (where a thread can "float").

Affinity (list inside a place)

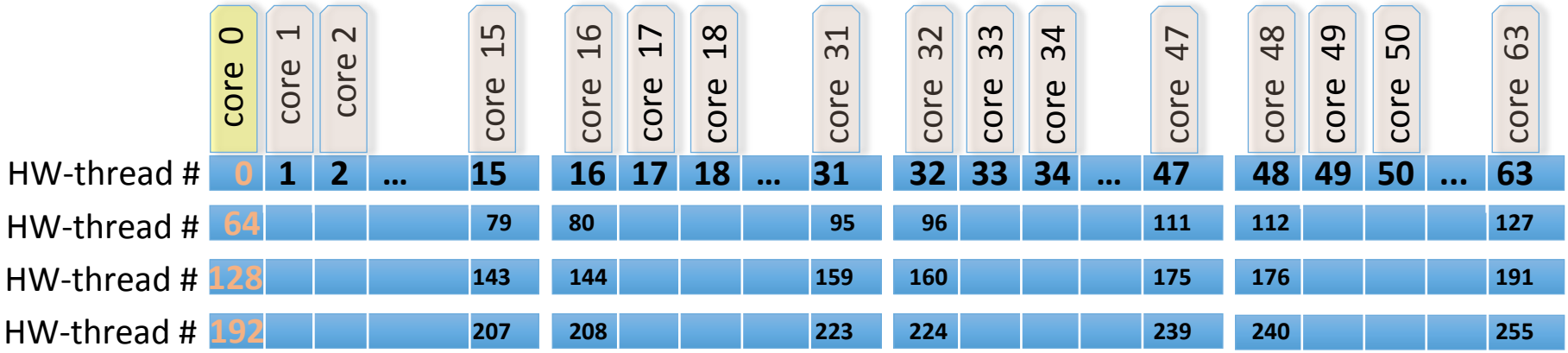
- A list inside a place creates a mask:

  Comma separated list e.g. { 0, 1, 2, 3, 4, 5 }

- Basic list syntax: *{ lower-bound : length : stride }*

  { lb : length } e.g. {0:4} = {0, 1, 2, 3}
  { lb : length : stride } e.g. {0:3:2} = {0, 2, 4}

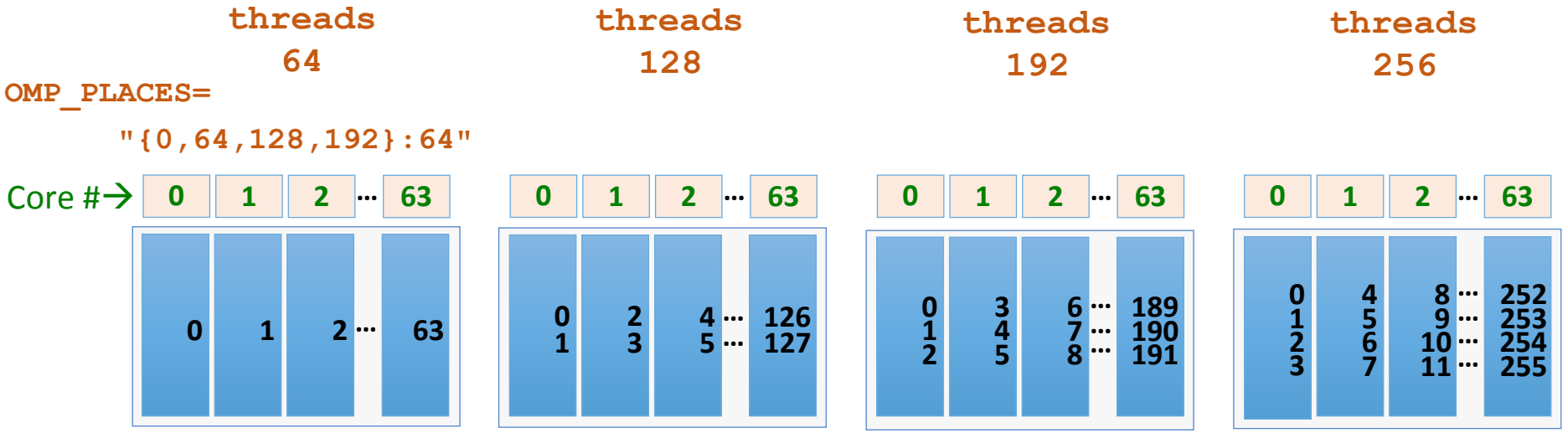
Hardware Threads



Layout is for a 64-core system

Affinity (cores)

List in a place allows a thread to float on any HW-thread (in a core here).



Summary

- When using all KNL HW-threads, just set the number of threads needed– that may work fine.
- Try using OMP_PROC_BIND=spread for sequential thread ids on a core (for 1, 2, 3 and 4 threads per core).
- Use OMP_PLACES for complex layouts of threads

  OMP_PLACES={*lower_bound**}:*length*:*stride*

  syntax is a compact form for building HW-thread lists.

* {lb} can be expanded as {*lb*:*length*:*stride*}

The End

See me after the workshop for questions.

Affinity (core summary)

- Affinity settings to assign 1, 2, 3 and 4 threads on a core.
- Threads in a core can execute on any HW-thread.
- Thread numbering is sequential on core.

OMP_ NUM_ THREADS	OMP_PLACES	Cores 0-15
64	{0,64,128,192}:64	1 thread/core
128	{0,64,128,192}:64	2 threads/core
192	{0,64,128,192}:64	3 thread/core
256	{0,64,128,192}:64	4 threads/core

OMP_PROC_BIND does not need to be set.

```
e.g. export OMP_NUM_THREADS=128  OMP_PLACES="{0,64,128,192}:64"
```

Affinity (working with cores)

- Multiple locations can be included in a place.
- The list syntax is the same as the place syntax:

  { lower_bound : size : stride }

```
e.g.   {0:4:2}              = {0,2,4,6}
       {0:4:2},{1:4,2}      = {0,2,4,6},{1:3,5,7}
```

- Place lists can be replicated

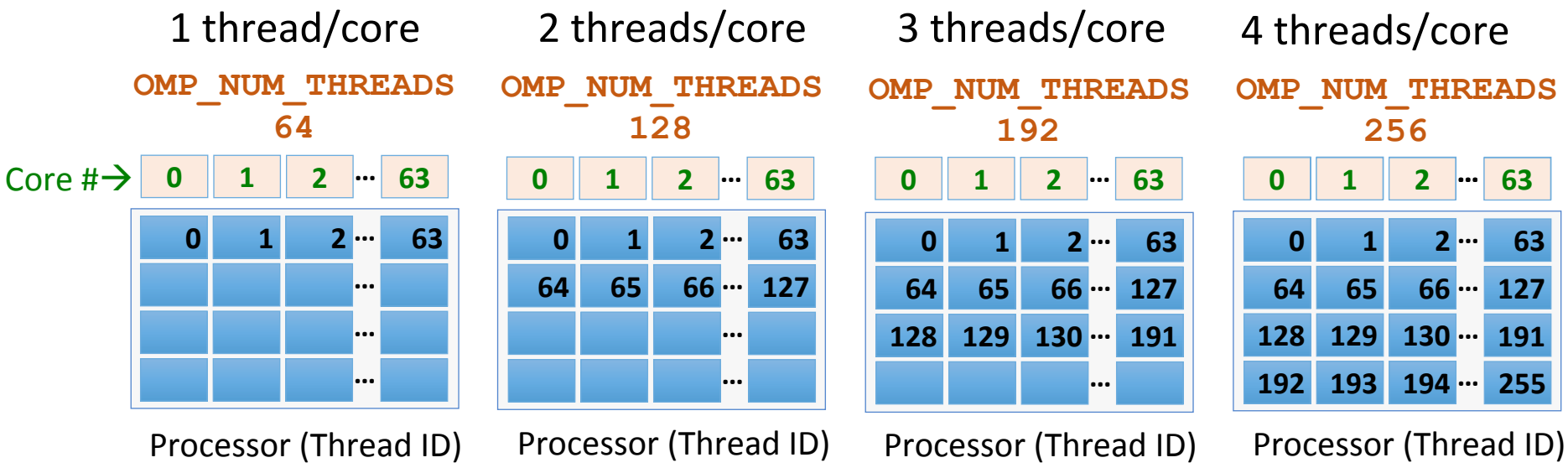
```
{0,64,128,192}:64             = 64 places of cores
                              = {0,64,128,192},{1,65,129,193},…

{0,64,128,192}:32:2           = 32 places of every other core
                              = {0,64,128,192},{2,66,130,194},…

{0:2,64:2,128:2,192:2}:32:2 = 32 places of tiles
                              = {0,1,64,65,128,129,192,193},
                                {2,3,66,67,130,131,194,195}, …
```

Default Affinity (distribution)

Occupying 1, 2, 3 and 4 HW threads per core:
(Default: thread ID # → HW-thread id)



This is OK for some applications, but:

1.) Putting sequential threads on the same core may be more cache friendly,

2.) Allowing core threads to "float" in the core may be more balance friendly.

Affinity (cores)

List in a place allows a thread to float on any HW-thread (in a core here).