IWOMP05 panel
"OpenMP 3.0"

Mitsuhisa Sato

(University of Tsukuba, Japan)

Final comments

in EWOMP03 panel “What are the necessary ingredients for scalable OpenMP programming”

- Performance of OpenMP for SDSM
  - good for some applications, but sometimes bad
  - it depends on network performance.
- We should look at PC-clusters
  - High performance and good cost-performance
    - “will converge to cluster of small SMP nodes” (by Tim@EWOMP 2001)
  - Large scale SMPs can survive?
- Mixed OpenMP-MPI does not help unless you already have MPI code.
  - “Life is too short for MPI” (by T-shirts message@WOMPAT2001)
- We should learn from HPF.

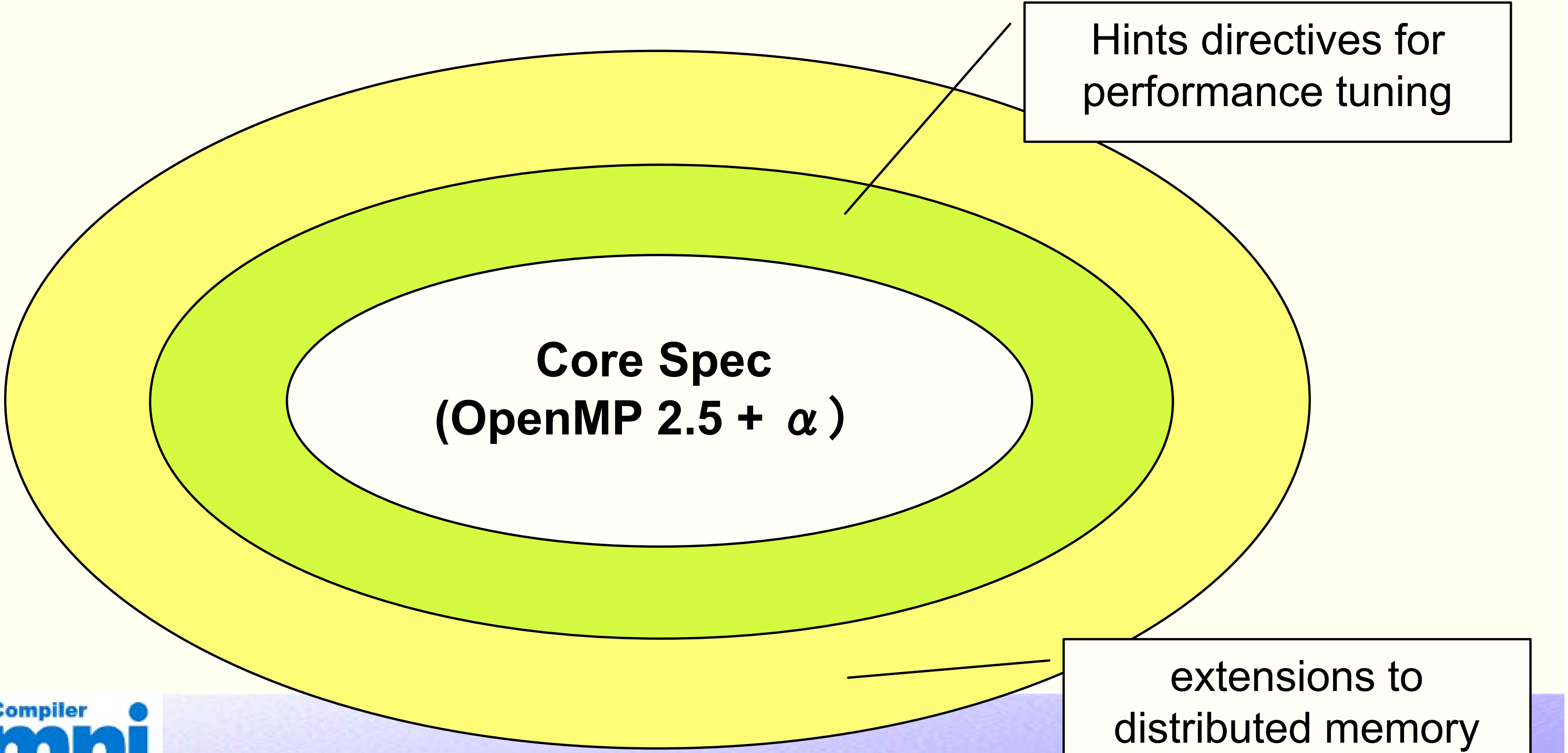
Many idea and proposals, so far ….

- Task queue construct (by KAI)
- conditional variable in critical construct
- processor-binding
- nested parallelism, multi-dimensional parallel loop
- post/wait in sections construct (task-level parallelism) (by UPC?)
- For DSM
  - next touch
  - mapping directives & affinity scheduling for loop (Omni/SCASH)
- "Threadshared" in Cluster OpenMP (KAI?)
- ….

- OpenMP on Software DSM for distributed memory
  - Very attractive, but …
  - Limitation of shared memory model for a large-scale system (100～ processors)
    - Requires a large single address (naming) space to map the whole data.
    - may require large amount of memory and TLBs ….

For OpenMP3.0

- Core spec. to define programming model
- Hints directives for performance tuning (esp. for DSM)
- Extensions to distributed memory



Hints directives for performance tuning

Core Spec
(OpenMP 2.5 + α)

extensions to distributed memory

OpenMP3.0 Core Spec

- Core spec to define programming model
  - mandatory spec. to be compliant
  - OpenMP 2.5 + α
  - Candidates (α) may include:
    - Task queue construct (by KAI)
    - conditional variable in critical construct
    - processor-binding
    - nested parallelism, multi-dimensional parallel loop
    - post/wait in sections construct (task-level parallelism)

Hint directives

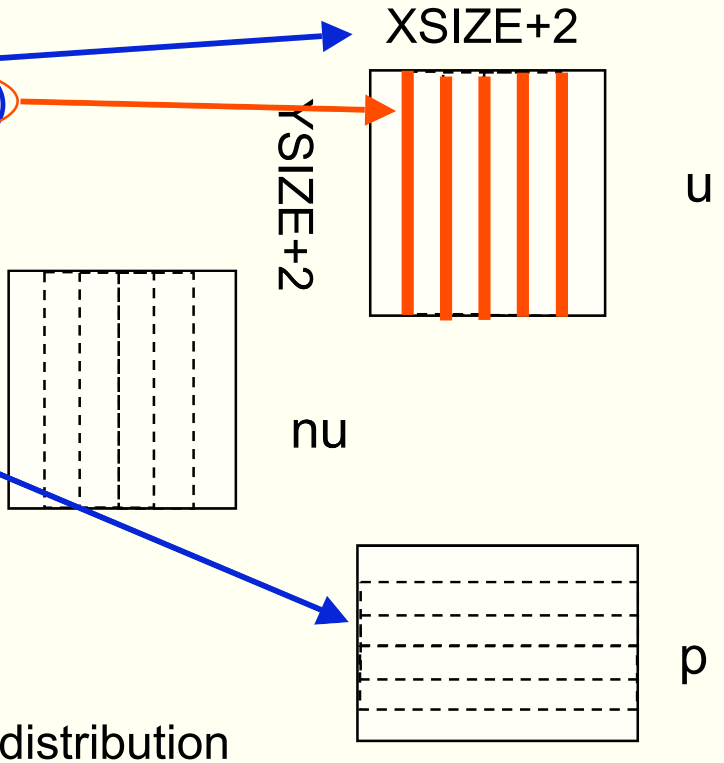
- For performance tuning
  - **Performance is a key for HPC!**
  - Not mandatory
    - it can be ignored
  - May include:
    - To exploit locality (esp. for Hardware/Software DSM)
      - next touch/first touch
      - mapping directives & affinity scheduling for loop
    - For better (loop) scheduling
      - …? ….

Extensions for distributed memory

- “We should look at PC cluster (distributed memory)”
  - Everybody says “OpenMP is good, but no help for cluster…”
- Should be defined outside of OpenMP
  - may be nested with OpenMP core spec.
    - Inside node, OpenMP core spec.
    - outside node, use the extensions
- Candidates will be:
  - “Threadshared” in Cluster OpenMP by KAI
    - “private” is default. “shared” must be specifed.
  - UPC
  - CAF
  - (HPF?, too much!?)
  - We have proposed “OpenMPI” (not Open MPI!) in the last EWOMP

An example of “OpenMPI”

```
#pragma ompi distvar (dim=1,sleeve=1)
double u[YSIZE + 2][XSIZE + 2];
#pragma ompi distvar (dim=1)
double nu[YSIZE + 2][XSIZE + 2];
#pragma ompi distvar (dim=0)
int p[YSIZE + 2][XSIZE + 2];
................
#pragma ompi for
    for(j = 1; j <= XSIZE; j++)
      u[i][j] = 1.0;
#pragma ompi for
  for(j = 1; j <= XSIZE; j++){
    u[0][j] = 10.0;
    u[YSIZE+1][j] = 10.0;
  }
```



- Array distribution
- Data consistency
  - With “sleeve” notation, necessary data are exchanged among neighboring processes
- Data reduction
- Data synchronization
  - Pseudo global variables should be synchronized

OpenMP for distributed Memory?

- Limitation of shared memory model for very large-scale system (100～ processors)
  - Requires a large single address (naming) space to map the whole data.
  - may require large amount of memory and TLBs ….
  - 64 bit address space is required.
- Distributed Array like in HPF
  - A portion of array is stored into each processor.
  - It is different from uniform shared memory address space
    - OK in Fortran, but NG in C.
  - Mixed HPF-OpenMP?
  - OpenMP extension like HPF?
  - …
- OpenMP should learn from HPF !?