

OpenMP Does Not Scale

- What do we mean by good scaling?
 - 75% efficiency?
 - 8CPUs?
- Who cares?
 - Large SSI machines are not so common
- Individual resources:
 - Generally ≤ 8 CPUs
 - Costs strongly biases choice (10*dual CPU v 1*8CPU box)
- Local resources:
 - Maybe few 16-32CPU shared memory systems
 - What is reasonable allocated fraction?
- Remote shared:
 - Few installations have > 32 CPU systems
 - Main contenders Sun, SGI, Fujitsu, HP(?),

OpenMP Does Not Scale

- What do we mean by good scaling?

75% efficiency?

Resource availability and allocation thereof forces users to distributed memory paradigms

- Large SSI machines are not so common
- Individual resources:

Scaling of OpenMP is not the issue(?)

- Local resources:
 - Maybe few 16-32CPU shared memory systems

Justifying time to develop scalable OpenMP applications is the issue

- Main contenders Sun, SGI, Fujitsu, HP(?),

Motivating the Development of Scalable OpenMP Applications

OpenMP and Clusters

OpenMP and Clusters

- Hybrid OpenMP and MPI
 - Need MPI implementation
 - Why not just use MPI
- Distributed shared memory OpenMP
 - Poor performance?
- OpenMP overlaid on Global Arrays
 - Technically possible

OpenMP and Clusters

?Hybrid OpenMP and Global Arrays?

- Implications for OMP?

- Any better than OMP/MPI?

- Distributed shared memory OpenMP

- Poor performance?

?What extensions are required to make OMP run well under DSM?

- Technically possible

Some Suggested OMP 3.0 Extensions

Compute Centrix	Data Centrix
OMP_SET_THREAD_NUM	OMP_SET_MLD_NUM
OMP_GET_NUM_THREAD	OMP_GET_NUM_MLD
OMP_GET_THREAD_NUM	OMP_GET_MLD(ADDRESS)
	OMP_RUN_ON_MLD
malloc	malloc_on_mld()

Some Suggested OMP 3.0 Extensions

And how about a global counter..

```
mycounter=omp_init_counter(value,chunk)
```

```
next_value=omp_next_counter(mycounter)
```

OMP_RUN_ON_MLD

malloc

malloc_on_mld()

2-D Heat Diffusion

Exp No.	C-Brick Pair 1		C-Brick Pair 2		Best Sec
	Thread	Memory	Thread	Memory	
2.1	0	0	0	0	3.2
2.2	0	0	1	1	3.1
2.3	0	1	1	0	4.0
2.4	9	0	9	0	4.6
2.5	9	0	10	1	4.4
2.6	IRIX	IRIX	IRIX	IRIX	4.3