

Communication and Optimization Aspects of Parallel Programming Models on Hybrid Architectures

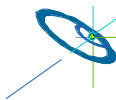
Rolf Rabenseifner
rabenseifner@hlrs.de

Gerhard Wellein
gerhard.wellein@rrze.uni-erlangen.de

University of Stuttgart
High Performance
Computing Center Stuttgart
HLRS
www.hlrs.de

University of Erlangen
Regionales Rechenzentrum
RRZE

EWOMP 2002, Rome, Italy, Sep. 18–20, 2002

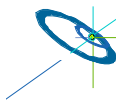


Cluster Programming Models
Slide 1
Höchstleistungsrechenzentrum Stuttgart

H L R I S

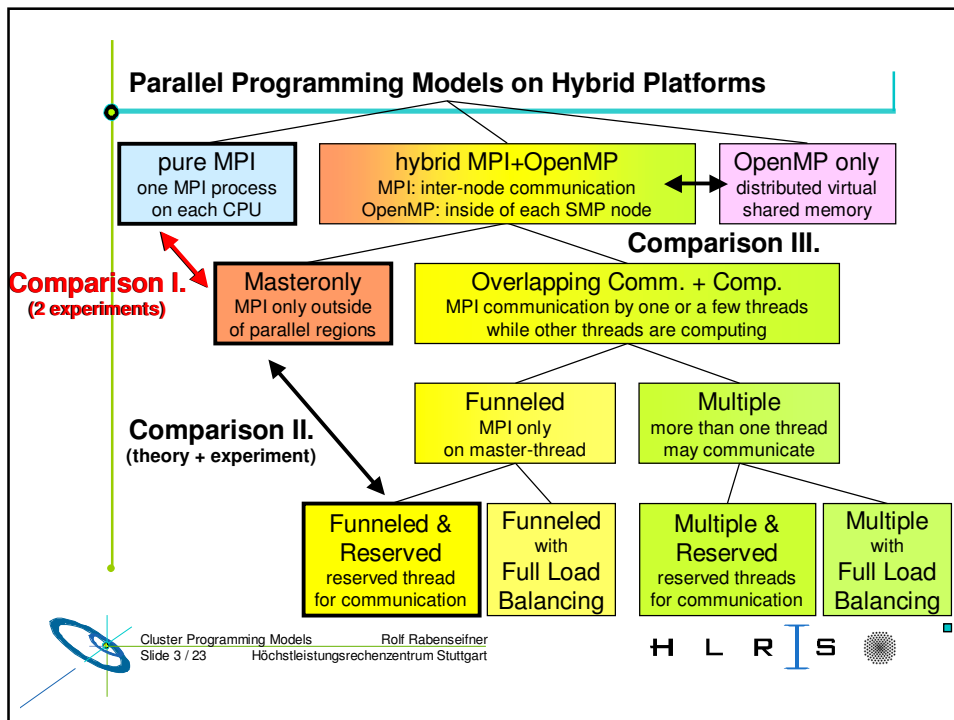
Motivation & Goals

- HPC systems
 - often clusters of SMP nodes = hybrid architectures
 - Often **hybrid programming (MPI+OpenMP)** slower than **pure MPI**
 - why?
 - Using the communication bandwidth of the hardware
 - Minimizing synchronization = idle time
- } **optimal usage of the hardware**
- Appropriate parallel programming models
 - Pros & Cons
 - Work horses for legacy & new parallel applications
 - Optimization of the middleware



Cluster Programming Models
Slide 2 / 23
Rolf Rabenseifner
Höchstleistungsrechenzentrum Stuttgart

H L R I S



MPI + OpenMP

- MPI_Init_threads(required, &provided) MPI 2.0: provided=
- categories of thread-safety: MPI_THREAD_...
- no thread-support by MPI ..._SINGLE
- MPI process may be *sometimes* multi-threaded, ... ~~SINGLE~~
- (parallel regions) and MPI is called only if only the master-thread exists
- Same, but the other threads may sleep ... ~~SINGLE~~
- MPI may be called ... ~~SINGLE~~
- only outside of OpenMP parallel regions ... **MASTERONLY** { proposal for MPI 2.1
- Same, but all other threads may compute ..._FUNNELED
- Multiple threads may call MPI, but only one thread may execute an MPI routine at a time ..._SERIALIZED
- MPI may be called from any thread ..._MULTIPLE

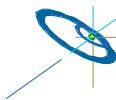
Cluster Programming Models Rolf Rabenseifner
Slide 4 / 23 Höchstleistungsrechenzentrum Stuttgart

HLRIS

MPI + OpenMP

pure MPI
Masteronly

- using OMP MASTER → MPI_THREAD_FUNNELED needed
 - no implied barrier !
 - no implied cache flush !
- using OMP SINGLE → MPI_THREAD_SERIALIZED needed
- $A[i] = \dots$
 OMP MASTER / SINGLE < OMP BARRIER
 MPI_Send(A, ...)
 OMP END MASTER / SINGLE < OMP BARRIER
 $A[i] = \text{new value}$
- Same problem as with MPI_THREAD_MASTERONLY:
 - all application threads are sleeping while MPI is executing



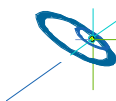
Cluster Programming Models Rolf Rabenseifner
Slide 5 / 23 Höchstleistungsrechenzentrum Stuttgart

H L R I S

Pure MPI on hybrid architectures

pure MPI
Masteronly

- Optimizing the communication
 - best ranking of MPI processes on the cluster
 - based on MPI virtual topology
 - sequential ranking: 0, 1, 2, 3, ... 7 8, 9, 10, ... 15 16, 17, 18 ... 23
 - round robin: 0, N, 2N ... 7N 1, N+1, ... 7N+1 2, N+2, ... 7N+2
- (Example: N = number of used nodes, 8 threads per node)



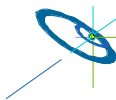
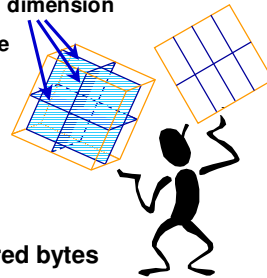
Cluster Programming Models Rolf Rabenseifner
Slide 6 / 23 Höchstleistungsrechenzentrum Stuttgart

H L R I S

Pure MPI on hybrid architectures (continued)

pure MPI
Masteronly

- Additional message transfer inside of each node
 - compared with MPI+OpenMP
 - Example: 3-D (or 2-D) domain decomposition
 - e.g. on 8-way SMP nodes
 - one (or 1–3) additional cutting plane in each dimension
 - expecting same message size on each plane
 - outer boundary (pure MPI)
 - inner plane (pure MPI)
 - outer boundary (MPI+OpenMP)
- pure MPI compared with MPI+OpenMP:
 - only doubling the total amount of transferred bytes



Cluster Programming Models Rolf Rabenseifner
Slide 7 / 23 Höchstleistungsrechenzentrum Stuttgart

H L R I S

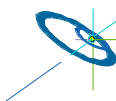
Benchmark results

pure MPI
Masteronly

- On Hitachi SR8000, b_eff¹⁾ benchmark on 12 nodes

	b_eff	b_eff Lmax ²⁾	3-d-cyclic average	3-d-cyclic Lmax ²⁾
aggregated bandwidth – hybrid [MB/s]	1535	5565	1604	5638
(per node) [MB/s]	(128)	(464)	(134)	(470)
aggregated bandwidth – pure MPI [MB/s]	5299	16624	5000	18458
(per process) [MB/s]	(55)	(173)	(52)	(192)
bw _{pure MPI} / bw _{hybrid} (measured)	3.45	2.99	3.12	3.27
size _{pure MPI} / size _{hybrid} (assumed)	2 (based on last slide)			
T _{hybrid} / T _{pure MPI} (concluding)	1.73	1.49	1.56	1.64

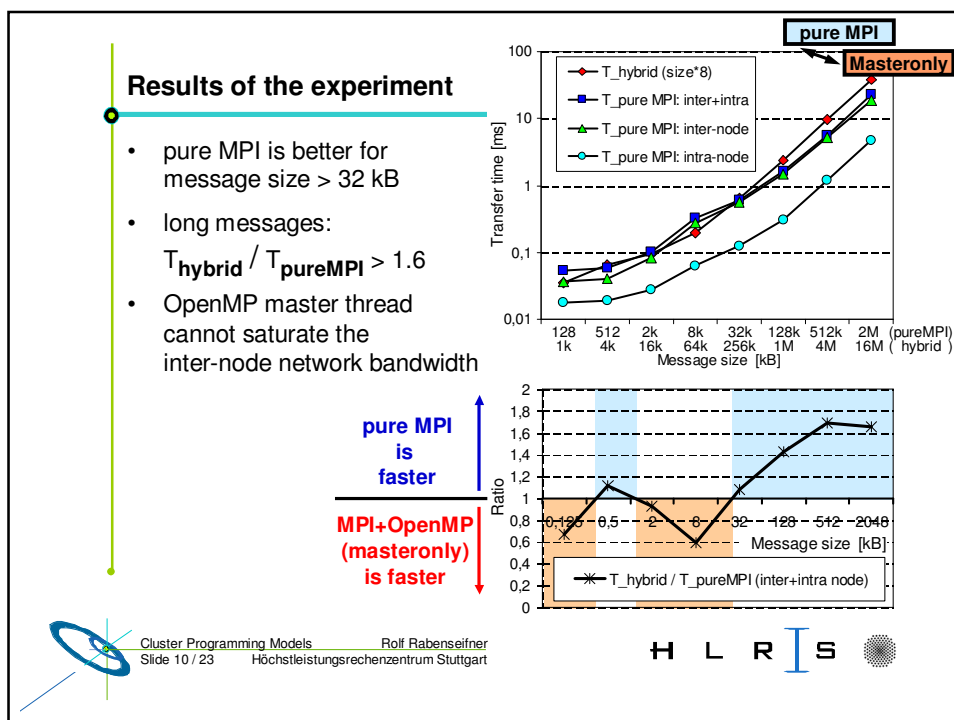
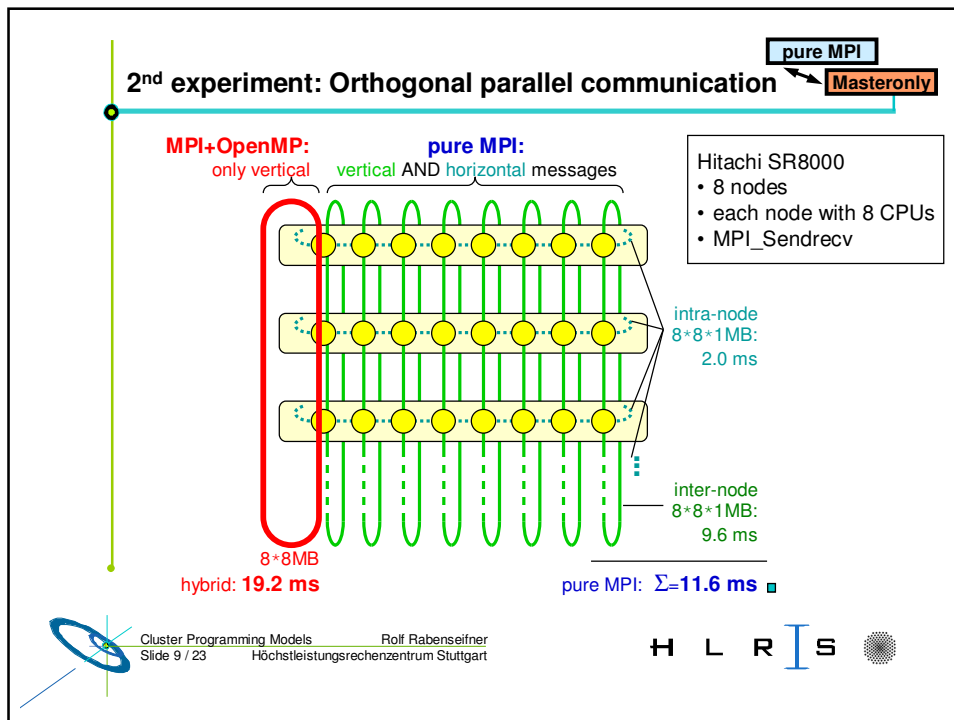
→ communication with pure MPI model is about 60% faster than with the hybrid-masteronly model



Cluster Programming Models Rolf Rabenseifner
Slide 8 / 23 Höchstleistungsrechenzentrum Stuttgart

H L R I S

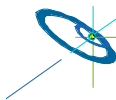
¹⁾ www.hlr.de/mpi/b_eff/ ²⁾ message size = 8MB



Interpretation of the benchmark results



- If the inter-node bandwidth cannot be consumed by using only one processor of each node
→ the pure MPI model can achieve a better aggregate bandwidth
- If $\text{bw}_{\text{pure MPI}} / \text{bw}_{\text{hybrid}} > 2$ & $\text{size}_{\text{pure MPI}} / \text{size}_{\text{hybrid}} < 2$
→ **faster communication with pure MPI**
- If $\text{bw}_{\text{pure MPI}} = \text{bw}_{\text{hybrid}}$ & $\text{size}_{\text{pure MPI}} > \text{size}_{\text{hybrid}}$
→ faster communication with hybrid MPI+OpenMP



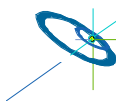
Cluster Programming Models Rolf Rabenseifner
Slide 11 / 23 Höchstleistungsrechenzentrum Stuttgart

H L R I S

Optimizing the hybrid **masteronly** model



- By the MPI library:
 - Using multiple threads
 - using multiple memory paths (e.g., for strided data)
 - using multiple floating point units (e.g., for reduction)
 - using multiple communication links (if one link cannot saturate the hardware inter-node bandwidth)
 - requires knowledge about free CPUs, e.g., via new `MPI_THREAD_MASTERONLY`
- By the user-application:
 - unburden MPI from work, that can be done by the application
 - e.g., concatenate strided data in parallel regions
 - reduction operations (`MPI_reduce` / `MPI_Allreduce`):
 - numerical operations by user-defined multi-threaded call-back routines
 - no rules in the MPI standard about multi-threading of such call-back routine



Cluster Programming Models Rolf Rabenseifner
Slide 12 / 23 Höchstleistungsrechenzentrum Stuttgart

H L R I S

pure MPI

Masteronly

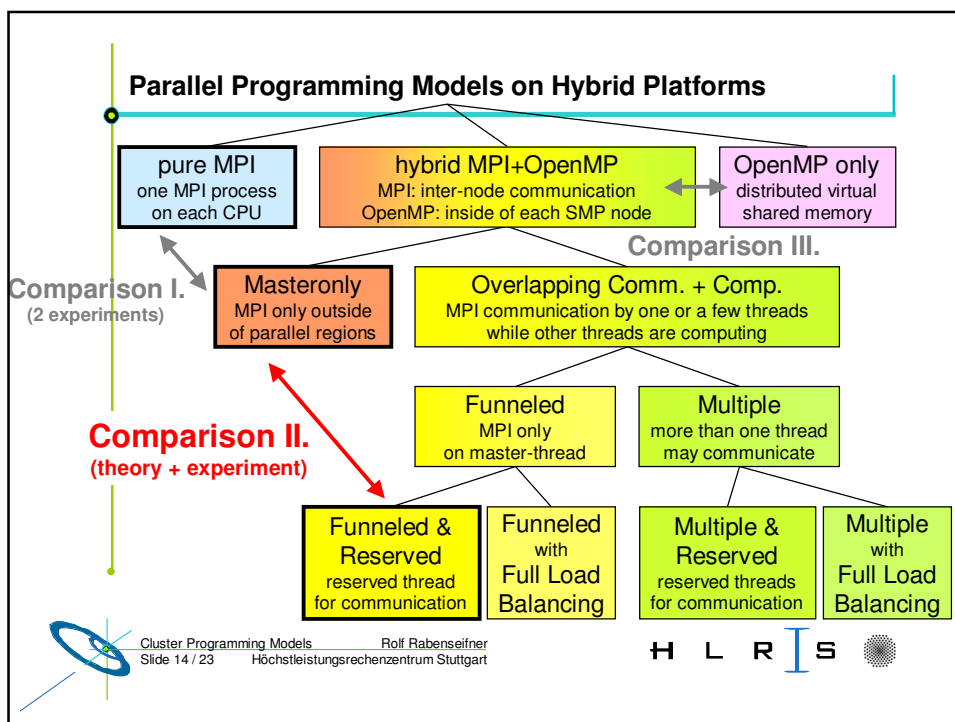
Other Advantages of Hybrid MPI+OpenMP

- No communication overhead inside of the SMP nodes
- Larger message sizes on the boundary
 - reduced latency-based overheads
- Reduced number of MPI processes
 - better speedup (Amdahl's law)
 - faster convergence, e.g., if multigrid numeric is computed only on a partial grid

Cluster Programming Models
Slide 13 / 23

Rolf Rabenseifner
Höchstleistungsrechenzentrum Stuttgart

HLRS



Masteronly

funneled & reserved

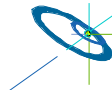
Overlapping computation & communication

The model: **communication funneled through master thread**

- Hybrid MPI+OpenMP
- Requires at least MPI_THREAD_FUNNELED
- While master thread calls MPI routines:
 - all other threads are computing!**

The implications:


- no communication overhead inside of the SMP nodes
- better CPU usage
 - although inter-node bandwidth may be used only partially
- 2 levels of parallelism (MPI and OpenMP):
 - additional synchronization overhead
- Major drawback: load balancing necessary
 - alternative: reserved thread for communication → next slide



Cluster Programming Models
 Slide 15 / 23

Rolf Rabenseifner
 Höchstleistungsrechenzentrum Stuttgart

H L R I S



Masteronly

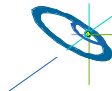
funneled & reserved

Overlapping computation & communication

Alternative:

- reserved tasks on threads:
 - master thread: communication
 - all other threads: computation
- cons:
 - bad load balance, if


$$\frac{T_{\text{communication}}}{T_{\text{computation}}} \neq \frac{n_{\text{communication_threads}}}{n_{\text{computation_threads}}}$$
- pros:
 - more easy programming scheme than with full load balancing
 - chance for good performance!

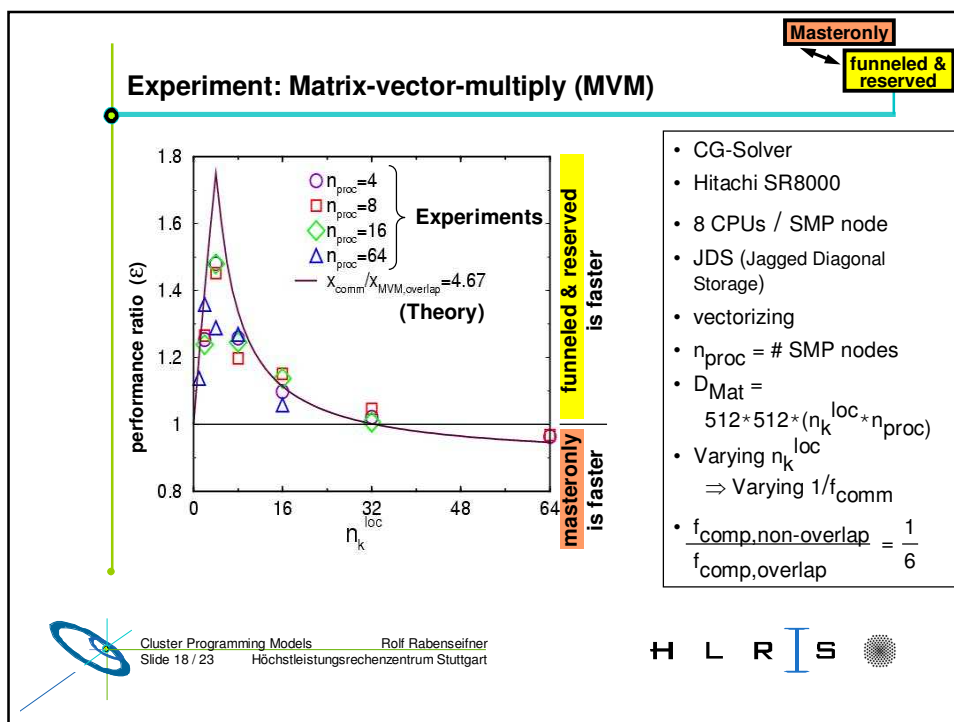
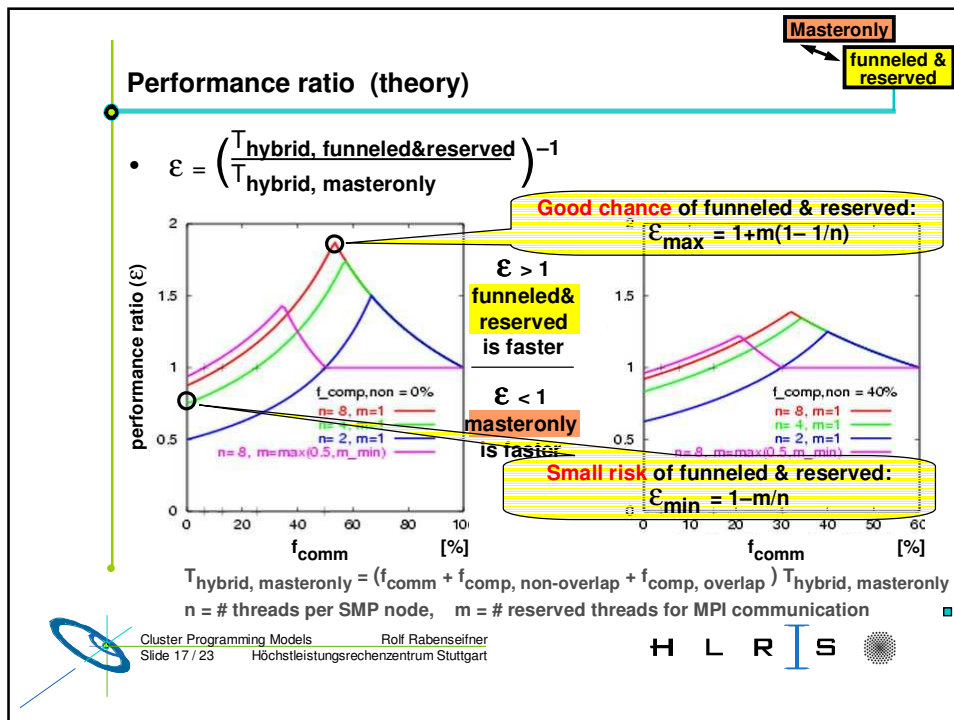


Cluster Programming Models
 Slide 16 / 23

Rolf Rabenseifner
 Höchstleistungsrechenzentrum Stuttgart

H L R I S





Comparison I & II – Some conclusions

pure MPI
one MPI process
on
each CPU

Comparison I.
(2 experiments)

hybrid MPI+OpenMP
Masteronly
MPI only outside of parallel
regions

Comparison II.
(theory + experiment)

Overlapping Comm. + Comp.
Funneled & Reserved
reserved thread
for communication

- One MPI thread should achieve full inter-node bandwidth
- For MPI 2.1:
MPI_THREAD_MASTERONLY
 - allows MPI-internal multi-threaded optimizations:
 - e.g., handling of derived data
 - reduction operations
- Application should overlap computation & communication
- Performance chance $\epsilon < 2$ (with one communication thread per SMP)
- ~50% performance benefit with real CG solver on Hitachi SR8000

Cluster Programming Models Rolf Rabenseifner
Slide 19 / 23 Höchstleistungsrechenzentrum Stuttgart

H L R I S

Parallel Programming Models on Hybrid Platforms

pure MPI
one MPI process
on each CPU

hybrid MPI+OpenMP
MPI: inter-node communication
OpenMP: inside of each SMP node

OpenMP only
distributed virtual
shared memory

Comparison III.

Comparison I.
(2 experiments)

Masteronly
MPI only outside
of parallel regions

Overlapping Comm. + Comp.
MPI communication by one or a few threads
while other threads are computing

Comparison II.
(theory + experiment)

Funneled
MPI only
on master thread

Multiple
more than one thread
may communicate

Funneled &
Reserved
reserved thread
for communication

Funneled
with
Full Load
Balancing

Multiple &
Reserved
reserved threads
for communication

Multiple
with
Full Load
Balancing

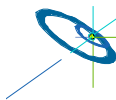
Cluster Programming Models Rolf Rabenseifner
Slide 20 / 23 Höchstleistungsrechenzentrum Stuttgart

H L R I S

Comparing other methods

Memory copies from remote memory to local CPU register and vice versa

Access method	Copies	Remarks	bandwidth $b(\text{message size})$
2-sided MPI	2	internal MPI buffer + application receive buf.	$b(\text{size}) = b_{\infty} / (1 + b_{\infty} T_{\text{latency}} / \text{size})$
1-sided MPI	1	application receive buffer	same formula, but probably better b_{∞} and T_{latency}
Compiler based: OpenMP on DSM (distributed shared memory) or with cluster extensions, UPC, Co-Array Fortran, HPF	1	page based transfer	extremely poor, if only parts are needed
	0	word based access	8 byte / T_{latency} , e.g., 8 byte / $0.33\mu\text{s} = 24\text{MB/s}$
	0	latency hiding with pre-fetch	b_{∞}
	1	latency hiding with buffering	see 1-sided communication



Compilation and Optimization

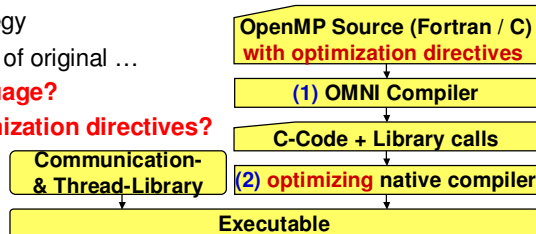
- Library based communication (e.g., MPI)
 - clearly separated optimization of
 - (1) communication → MPI library
 - (2) computation → Compiler

essential for success of MPI

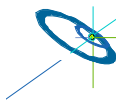
- Compiler based parallelization (including the communication):

- similar strategy
- preservation of original ...

- ... language?
- ... optimization directives?

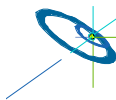


- Optimization of the computation more important than optimization of the communication**



Conclusions

- **Pure MPI** versus **hybrid masteronly** model:
 - Communication is bigger with pure MPI, but may be nevertheless faster
 - On the other hand, typically communication is only some percent → **relax**
- Efficient hybrid programming:
 - one should overlap communication and computation → **hard to do!**
 - using simple **hybrid funneled&reserved** model, you maybe **up to 50% faster** (compared to *masteronly* model)
- → Coming from **pure MPI**, one should try to implement **hybrid funneled&reserved model**
- If you want to use **pure OpenMP** (based on virtual shared memory)
 - try to use still the full bandwidth of the inter-node network (keep pressure on your compiler/DSM writer)
 - be sure that you do not lose any computational optimization
 - e.g., best Fortran compiler & optimization directives should be usable

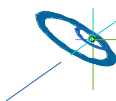


Cluster Programming Models Rolf Rabenseifner
Slide 23 / 23 Höchstleistungsrechenzentrum Stuttgart



Summary

- **Programming models on hybrid architectures** (clusters of SMP nodes)
 - Pure MPI / hybrid MPI+OpenMP masteronly / funneled-reserved / compiler based parallelization (e.g. OpenMP on clusters)
- **Communication**
 - difficulties with hybrid programming
 - multi-threading with MPI
 - bandwidth of inter-node network
 - overlapping of computation and communication, real chance: 50% performance benefit
 - latency hiding with compiler based parallelization
- **Optimization and compilation**
 - separation of optimization
 - we must not lose **optimization of computation**
- **Conclusion:**
 - Pure MPI → hybrid MPI+OpenMP → OpenMP on clusters
 - a roadmap full of stones!



Cluster Programming Models Rolf Rabenseifner
Slide 24 / 23 Höchstleistungsrechenzentrum Stuttgart

