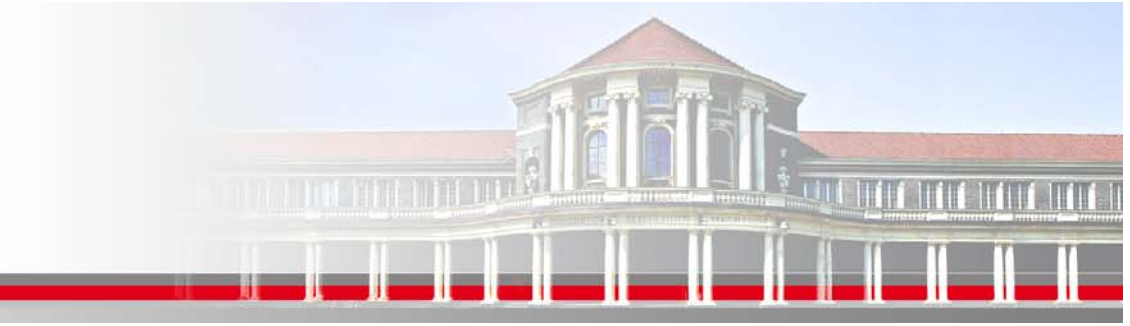
Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Praktikum: Paralleles Programmieren für Geowissenschaftler

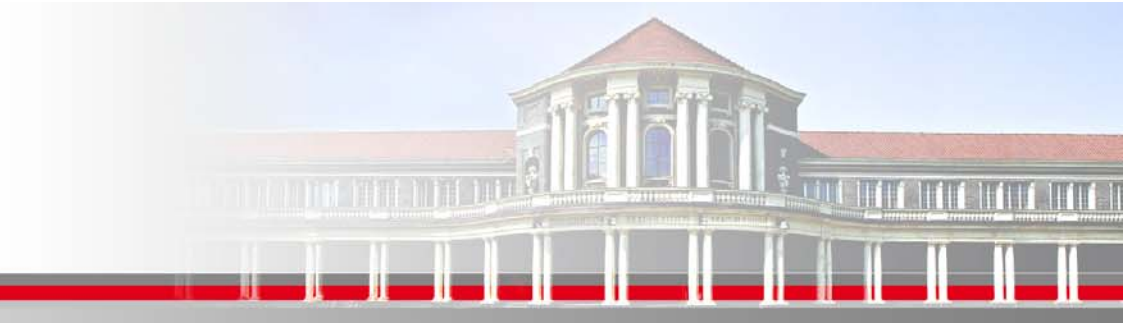
Prof. Thomas Ludwig, Hermann Lenhart & Enno Zickler

Dr. Hermann-J. Lenhart
hermann.lenhart@informatik.uni-hamburg.de



Übersicht:

- **Virtuelle Typologie**
- **Leistungsmessung – Speedup**
- **Übersicht an Punkten für die Note**



Virtuelle Topologie unter MPI

Die Beschreibung wie Prozesse in einem parallelen Rechner miteinander verbunden sind wird oft bezeichent als **Typologie** (topology of the computer)

In gängigen parallelen Programmen kommunizieren einzelne Prozesse meist nur mit wenigen anderen Prozessen;

Das Muster dieser Kommunikation wird bezeichnet als **„applied topology“** oder **„virtual typology“**

Virtuelle Topologie unter MPI

MPI erlaubt den Computerherstellern (Vendor)

eine Art „interner“ Optimierung durch die Implementierung von

MPI Topologie Funktionen (MPI topology functions)

MPI bildet damit die aktuelle Prozess-Typologie im Programm

auf die Typologie der internen Netzwerkkommunikation ab

Virtuelle Topologie unter MPI

MPI_CART_CREATE (MPI_COMM_WORLD, ndim, dims, isperiodic, &
reorder, comm2d, Ierror)

ndim = 2, z.B. dims(1) = 4 und dims(2) = 3 => Matrix 4x3

ferner

isperiodic(1) = .true. und isperiodic(2) = .true. => zyklische Randbedingungen

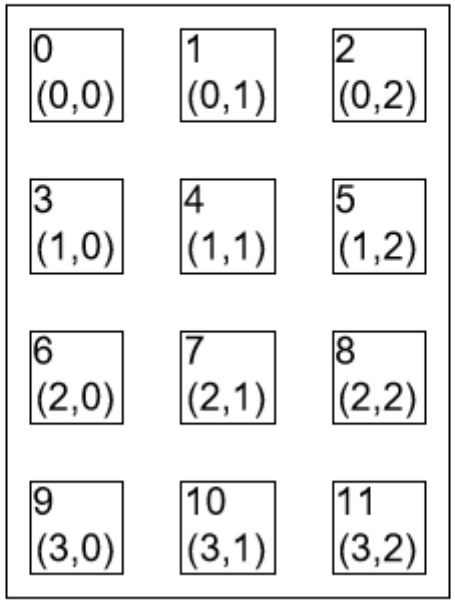
Innerhalb der neuen Topologie wird ein neuer Kommunikator comm2D

aus dem alten Kommunikator MPI_COMM_World gebildet

Reorder = .true. -> nächste Seite

Cartesian Topologies

4 x 3 cartesian grid

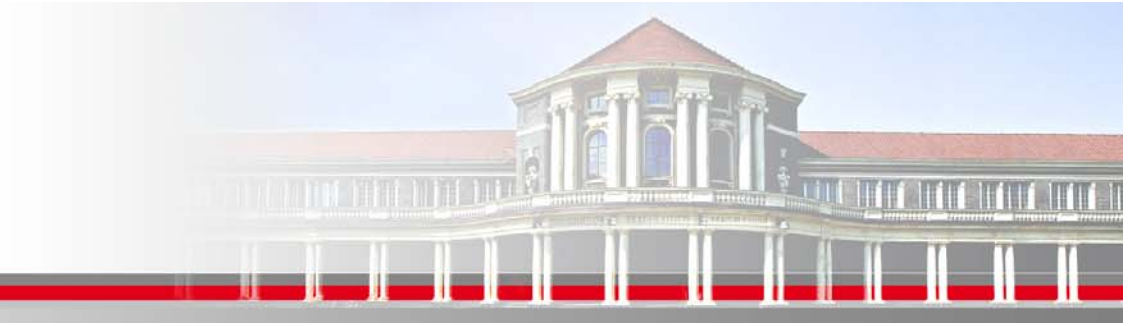


Es wurde nicht spezifiziert, welcher Prozess welchen Elementen in der Dekomposition der Matrix zugeordnet wird.

Reorder = .true.

erlaubt MPI intern die optimale Zuteilung bzw. Match der Prozesse auf die Elemente der Matrix.

Source: MPI Tutorial http://pl.postech.ac.kr/~gla/cs700-07f/ref/mpi/MPITutorial.pdf



Virtuelle Topologie unter MPI

Um sich durch diese virtuelle Typologie zu bewegen bietet MPI eigene Befehle:

Wichtig, all diese Befehle sind bezogen auf den neuen Kommunikator comm2d

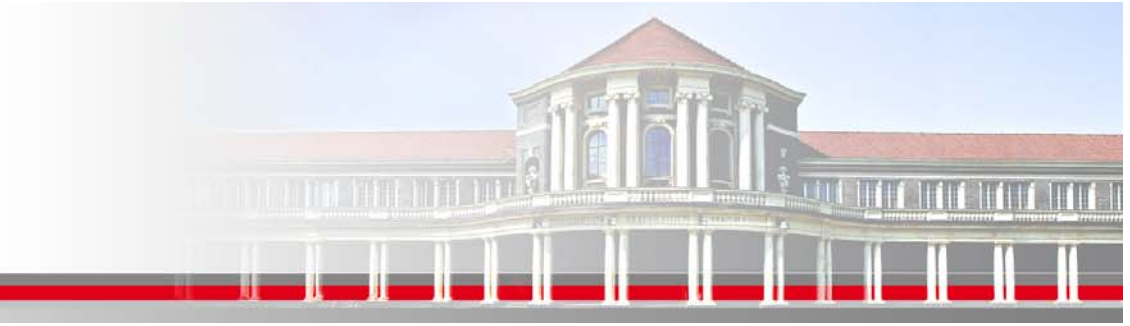
MPI_CART_GET (comm2d, ndim, dims, isperiodic, coord, Ierror)

Der MPI Routine enhält im Rückgabe die Wert:

ndim: die Dimension der Matrix

Coord: die kartesischen Koordinaten des aufrufenden Prozesses

isperioidc: die Vorgaben der Randwertbehandlung



Virtuelle Topologie unter MPI

MPI_CART_COORD (comm2d, rank , dim, coord, Ierror)

```
Call MPI_COMM_RANK (comm2d, myrank, ierror
Do rank = 0, n-1
Call MPI_CART_COORDS (comm2d, rank, dim, coord, ierror)
Write (*,*), rank , coord
Enddo
```

0,0 (0) 0,1 (1)

1,0 (2) 1,1 (3)

Virtuelle Topologie unter MPI

Die Abwicklung der Prozesse in Matrix wird abgebildet über die Befehle:

MPI_CART_SHIFT (comm2d, direction, displacement, src_rank, dest_rank)

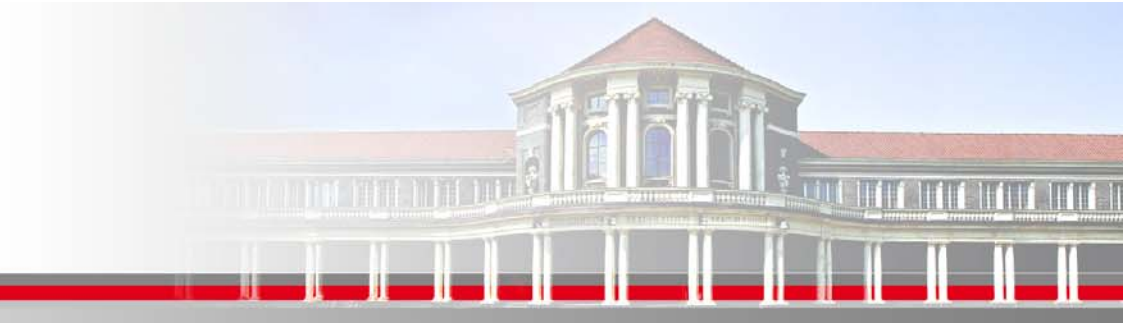
erfragt wer der Nachbar in dem System ist

z.B.

MPI_CART_SHIFT (comm2d, 0, 1, nbrtop, nbrbottom)

MPI_CART_SHIFT(comm2d, 1, 1, nbrleft, nbrright)

Diese Werte gehen in MPI_SENDRECV als Rang der Source und/oder Destination ein



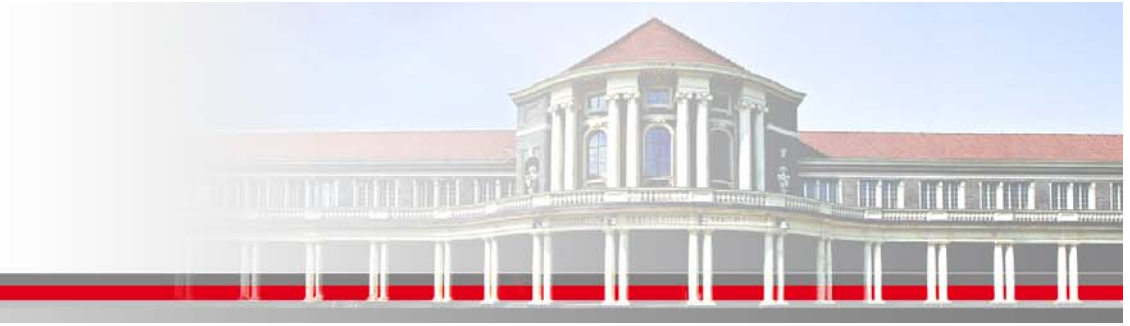
Virtuelle Topologie unter MPI

Zusammenfassung:

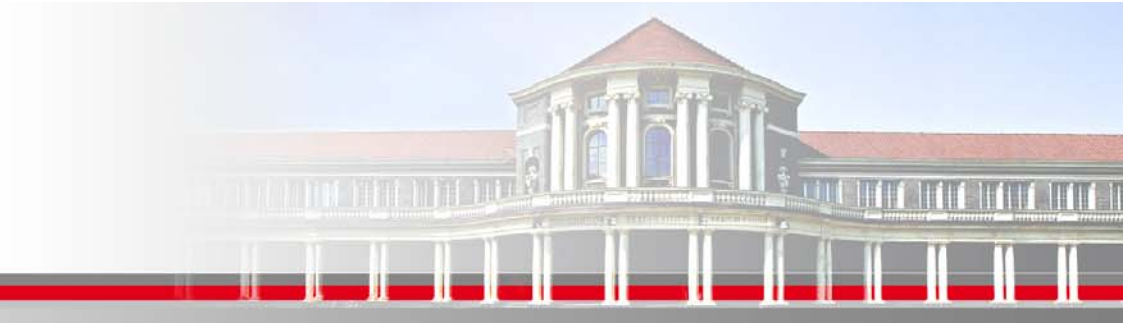
Zur Unterstützung der Performance von Programmen
bildet MPI die aktuelle Prozess-Typologie im Programm
auf die Typologie der internen Netzwerkkommunikation ab

Es werden unterstützt:

Kartesisches Gitter (mir und ohne zyklische Randbedingungen)
und andere Graphen



Leistungsmessung: Speedup



Speedup

Speedup (englisch für *Beschleunigung*) beschreibt mathematisch den Zusammenhang zwischen der seriellen und der parallelen Ausführungszeit eines Programmteils.

Der Speedup ist definiert durch die folgenden beiden Formeln:

$$S_p = \frac{T_1}{T_p}$$

wobei gilt:

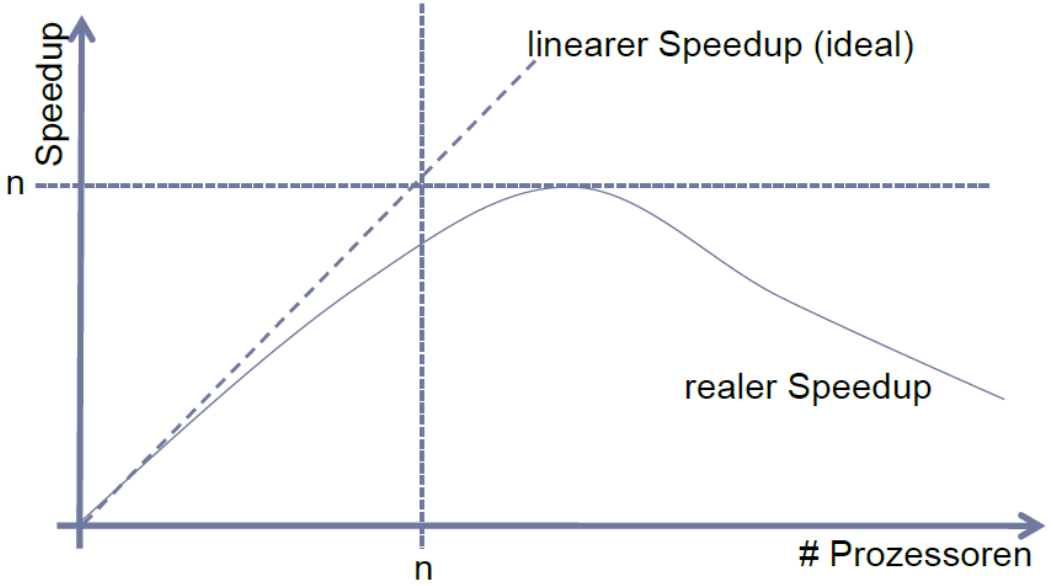
- p ist die Anzahl von Prozessoren
- S_p ist der theoretische Speedup, der erreicht werden kann bei Ausführung des Algorithmus auf p Prozessoren
- T_1 ist die Ausführungszeit auf einem Ein-Prozessor-System
- T_p ist die Ausführungszeit auf einem Mehrprozessorsystem

[Quelle: http://de.wikipedia.org/wiki/Speedup]

Speedup II

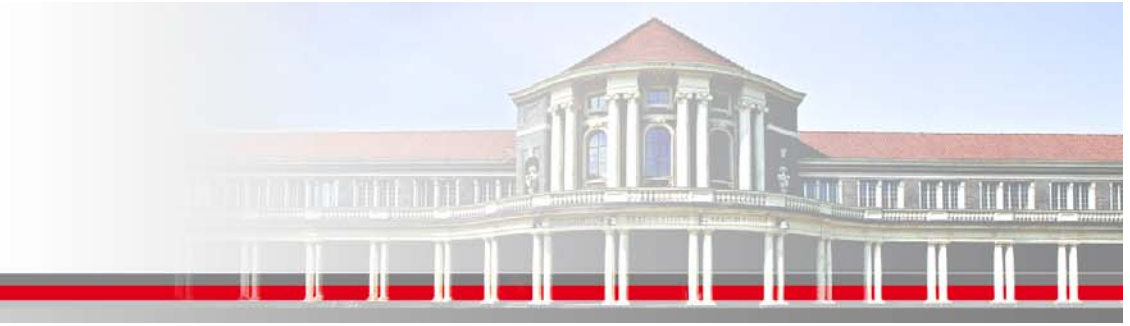
Darstellung der Speedup-Kurve

(nach Ludwig WS12/13)



Abweichung vom linearen Speedup:

- Anteil sequentiellem, nicht parallelisierbarem Programmcode -> Amdahlsches Gesetz
- Zunehmender Anteil an Kommuniktion bei steigender Anzahl an Prozessoren

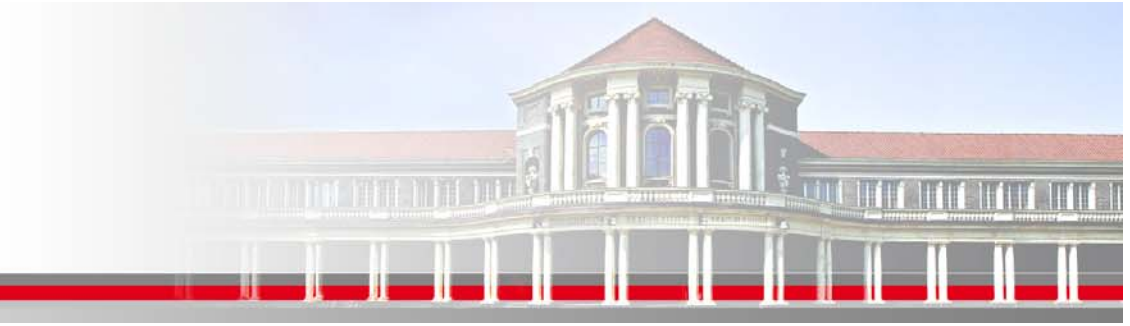


PPG Benotung:

Gesamtpunktzahl ohne Bonus
2300

50 % entspricht 1150 Punkte

Note	Prozentanteile
1.0	95-100 %
1.3	90-95 %
1.7	85-90 %
2.0	80-85 %
2.3	75-80 %
2.7	70-75 %
3.0	65-70 %
3.3	60-65 %
3.7	55-60 %
4.0	50 -55 %



Danke das wars!