



Universität Hamburg

DER FORSCHUNG | DER LEHRE | DER BILDUNG

Praktikum: Paralleles Programmieren für Geowissenschaftler

Prof. Thomas Ludwig, Hermann Lenhart & Tim Jammer



Dr. Hermann-J. Lenhart

hermann.lenhart@informatik.uni-hamburg.de



MPI Einführung III:

- ScatterV/ GatherV => nicht gleichmäßige Matrixaufteilung
- Halo Lines für Programmablauf

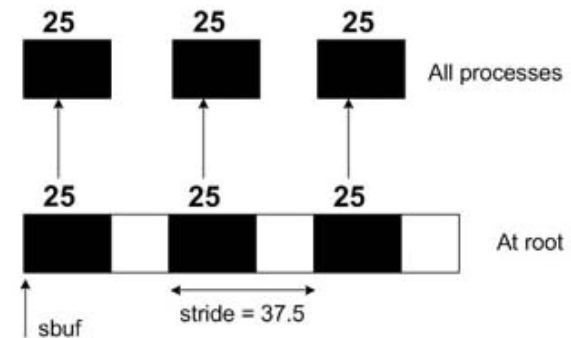
MPI Scatter-Gather: In der Literatur

MPI_Gatherv and MPI_Scatterv are the variable-message-size versions of MPI_Gather and MPI_Scatter.

MPI_Scatterv extends the functionality of MPI_Scatter to permit a varying count of data from each process.

MPI_SCATTERV Example

It does this by changing the **count** argument from a single integer to an integer array and providing a new argument **displs** (an array).





MPI Scatter -> ScatterV

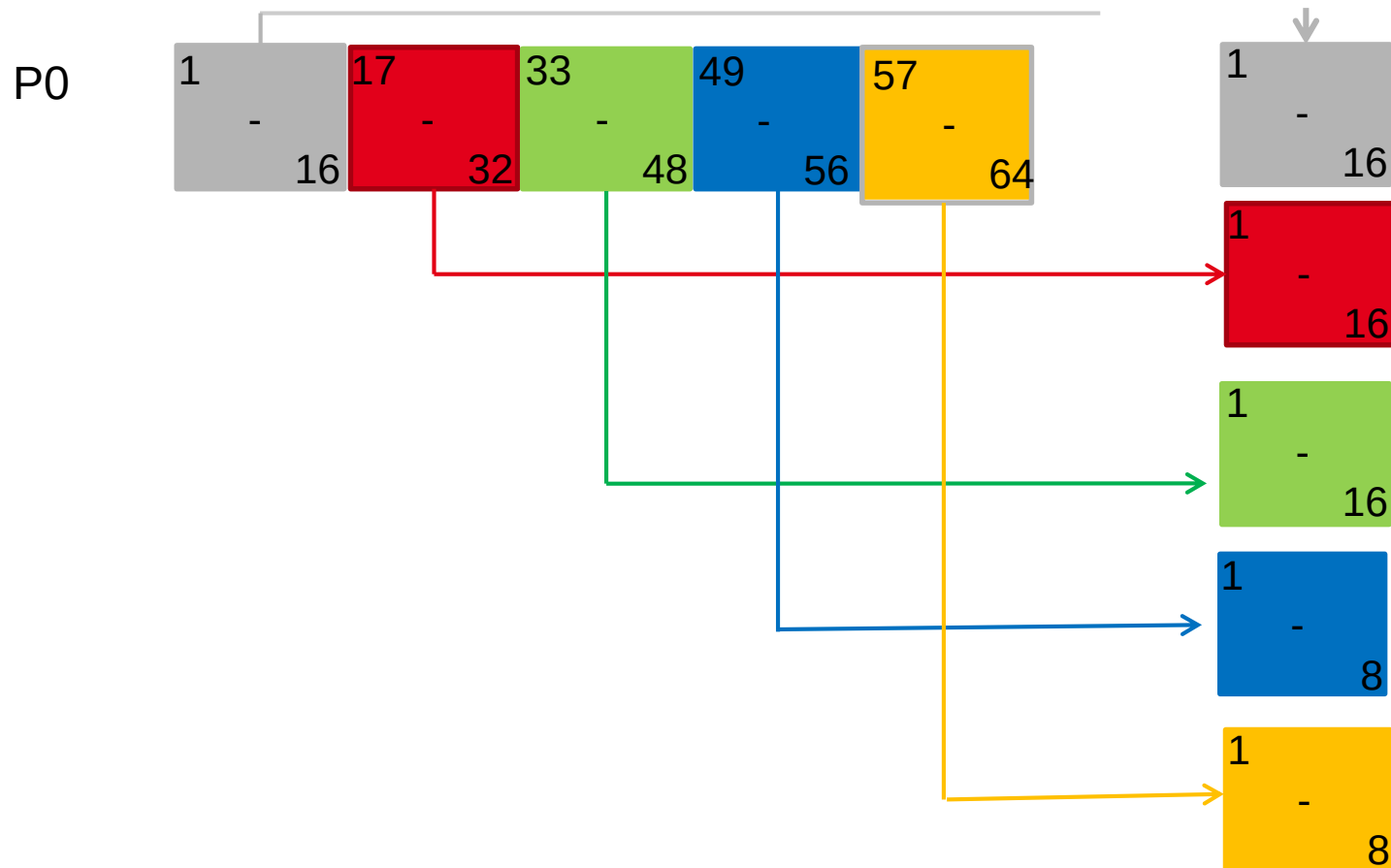
Call MPI_SCATTER (Send_Message, **Send_Count**, Send_Datatype,
Recv_Message, Recv_Count, Recv_Datatype,
Root, *Comm*, lerror)

Call MPI_SCATTERV(Send_Message, **Send_Count**, **Displacement**, Send_Datatype,
Recv_Message, Recv_Count, Recv_Datatype,
Root, *Comm*, lerror)

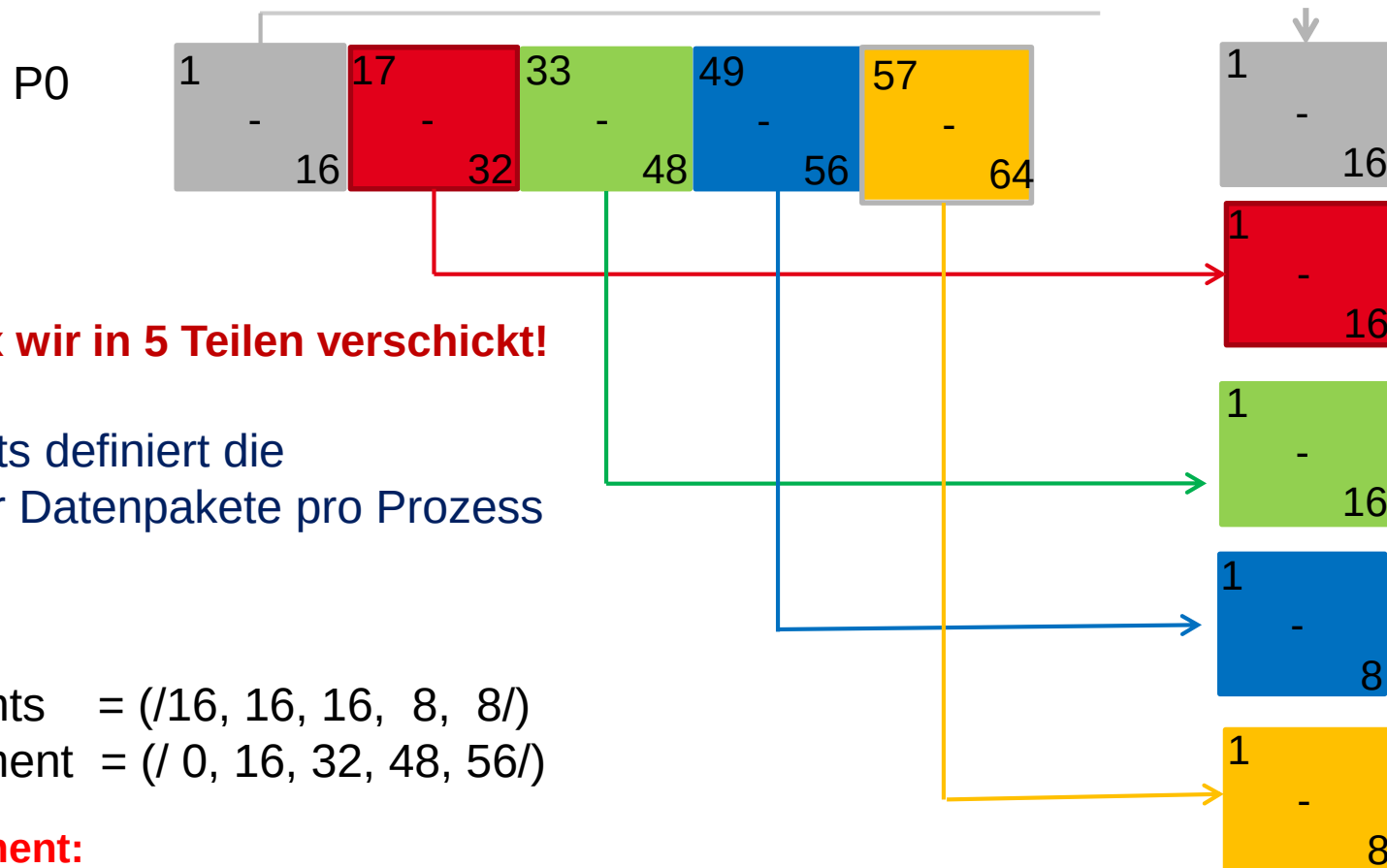
Änderung: **Send_count** von Integer - Zahl → Integer-Vektor

Neu: **Displacement-Vektor**

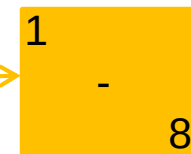
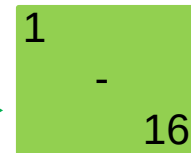
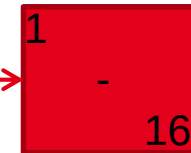
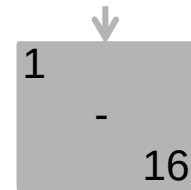
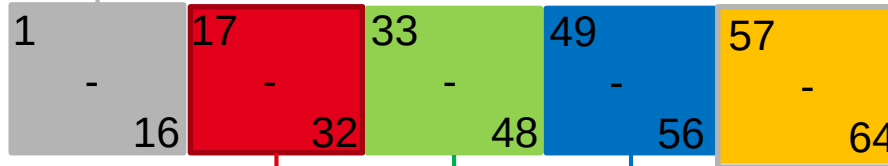
MPI ScatterV: Beispiel array2D(8X8) => 5 Teile chunk2D



MPI ScatterV: Beispiel array2D(8X8) => 5 Teile chunk2D



P0



Die Matrix wird in 5 Teilen verschickt!

sendcounts definiert die
Größe der Datenpakete pro Prozess

Sendcounts = (/16, 16, 16, 8, 8/)
displacement = (/ 0, 16, 32, 48, 56/)

Displacement:

Beginnt mit Null und bezieht sich jeweils auf das letzte Element des vorherigen Abschnitts!

MPI ScatterV / GatherV

! für 5 prozesse

```
integer, allocatable, dimension(8,8) :: array  
integer, allocatable, dimension(:, :) :: chunk, displacement, sendcounts
```

```
sendcounts      = (/16, 16, 16, 8, 8/)  
displacement    = (/ 0, 16, 32, 48, 56/)
```

```
call MPI_SCATTERV(array, sendcounts, displacement, MPI_INTEGER, chunk,  
sendcounts(rank+1), MPI_INTEGER, 0, MPI_COMM_WORLD, ierr)
```



MPI ScatterV / GatherV

! für 5 prozesse

```
integer, allocatable, dimension(8,8) :: array  
integer, allocatable, dimension(:, :) :: chunk, displacement, sendcounts
```

```
sendcounts      = (/16, 16, 16, 8, 8/)  
displacement    = (/ 0, 16, 32, 48, 56/)
```

```
call MPI_SCATTERV(array, sendcounts, displacement, MPI_INTEGER, chunk,  
sendcounts(rank+1), MPI_INTEGER, 0, MPI_COMM_WORLD, ierr)
```

```
call MPI_GATHERV(chunk, sendcounts(rank+1), MPI_INTEGER, array,  
sendcounts, displacement, MPI_INTEGER, 0, MPI_COMM_WORLD, ierr)
```

Bei GATHERV werden die Teilstücke entsprechend dem Rank zurückgeschickt!



Beispiel: Die Matrix wird in Vektoren zerlegt.

Für 10 Prozesse:

sendcounts = (/2, 10, 5, 8, 5, 2, 2, 2, 2, 2/)

displacement = (/0, 2, 12, 17, 25, 30, 32, 34, 36, 38/)

displacement def. Versatz in der Matrix

1	2								10
11	12					17			20
				25					30
	32		34		36		38		40



call MPI_SCATTERV(array, sendcounts, displacement, MPI_INTEGER,
chunk, sendcounts(rank+1), MPI_INTEGER, 0, MPI_COMM_WORLD, ierr)

Beispiel: Die Matrix wird in Vektoren zerlegt.

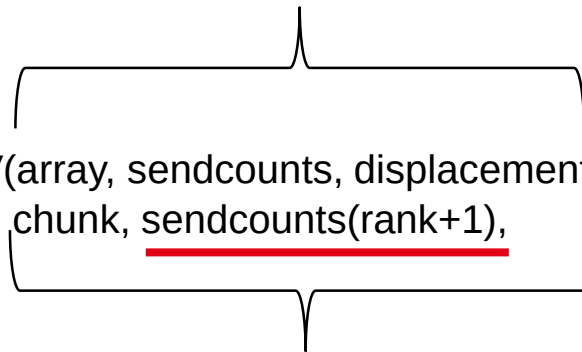
Für 10 Prozesse:

sendcounts = (/2, 10, 5, 8, 5, 2, 2, 2, 2, 2/)

displacement = (/0, 2, 12, 17, 25, 30, 32, 34, 36, 38/)

displacement def. Versatz in der Matrix

1	2								10
11	12					17			20
				25					30
	32		34		36		38		40



call MPI_SCATTERV(array, sendcounts, displacement, MPI_INTEGER,
chunk, sendcounts(rank+1), MPI_INTEGER, 0, MPI_COMM_WORLD, ierr)

Die Datenpakete werden den einzelnen Vektoren (chunk) pro Prozess zugewiesen

Entsprechend der variablen Größe pro Prozess definiert durch sendcount(rank+1)

Rank

0	1	2							
1	1	2	3	4	5	6	7	8	9
2	1	2	3	4	5				
3	1	2	3	4	5	6	7	8	
4	1	2	3	4	5				
5 x 9	1	2							

MPI Scatterv – Beispiel mit Lücke

1	2								10
11	12					17			20
				25					30
	32		34		36		38		40

Aufteilung Matrix 10x4

auf einzelne Vektoren

unterschiedlicher Länge

mit Lücke in den Daten

1	2
---	---

5 X

1	2
---	---

1	2	3	4	5	6	7	8		
---	---	---	---	---	---	---	---	--	--

1	2	3	4	5
---	---	---	---	---

1	2	3	4	5	6	7	8
---	---	---	---	---	---	---	---

1	2	3	4	5
---	---	---	---	---

1	2
---	---

Für 10 Prozesse:

integer :: sendcount (10)

Integer :: displacement (10)

1	2								10
11	12					17			20
				25					30
	32		34		36		38		40

1	2
---	---

1	2	3	4	5	6	7	8		
---	---	---	---	---	---	---	---	--	--

1	2	3	4	5
---	---	---	---	---

1	2	3	4	5	6	7	8
---	---	---	---	---	---	---	---

1	2	3	4	5
---	---	---	---	---

1	2
---	---

5 X

sendcounts = (/2, 8, 5, 8, 5, 2, 2, 2, 2, 2/)

displacement = (/0, 2, 12, 17, 25, 30, 32, 34, 36, 38/)

Sendcounts: wird an neue Datenstruktur angepasst, Position 11 und 12 wird ausgelassen

displacement: hier ändert sich in diesem Beispiel nichts.

MPI ScatterV / GatherV

Matrix 20 x 20

Zeilen	
6	A
	120
4	B
	200
3	C
	260
7	D
	400

Beispiel mit Teilmatrizen

A	B	C	D
sendcount = (/120,	80,	60,	140/)
displacement=(/ 0,	120,	200,	260/)



MPI ScatterV / GatherV

program scatterv

use mpi

integer, dimension(20, 20) :: array2D ! all the 2D data

integer, dimension(20, ?) :: chunk2D ! Max (!) piece of 2D data each process works on

integer :: mpi_ierr, mpi_rank, mpi_size, sendcnt(4), displ(4)

sendcnt = (/120,80,60,140/)

displ=(/0,120,200,260/)

CALL MPI_INIT(mpi_ierr)

CALL MPI_SCATTERV(array2D, sendcnt, displ, MPI_INTEGER, &
chunk2D, sendcnt(mpi_rank+1), MPI_INTEGER, 0, MPI_COMM_WORLD, mpi_ierr)

CALL computation(chunk2D,sendcnt(mpi_rank+1)) ! where the magic happens

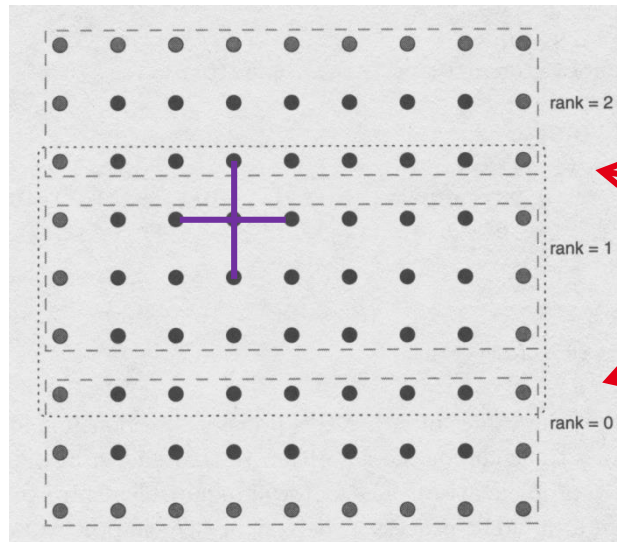
CALL MPI_GATHERV(chunk2D, sendcnt(mpi_rank+1), &
MPI_INTEGER, array2D, sendcnt, displ, MPI_INTEGER, 0, MPI_COMM_WORLD, mpi_ierr)

CALL MPI_FINALIZE(mpi_ierr)

end program scatterv



Parallele Bearbeitung einer Matrix I



Überlappung der Teilmatrizen
der einzelnen Prozesse

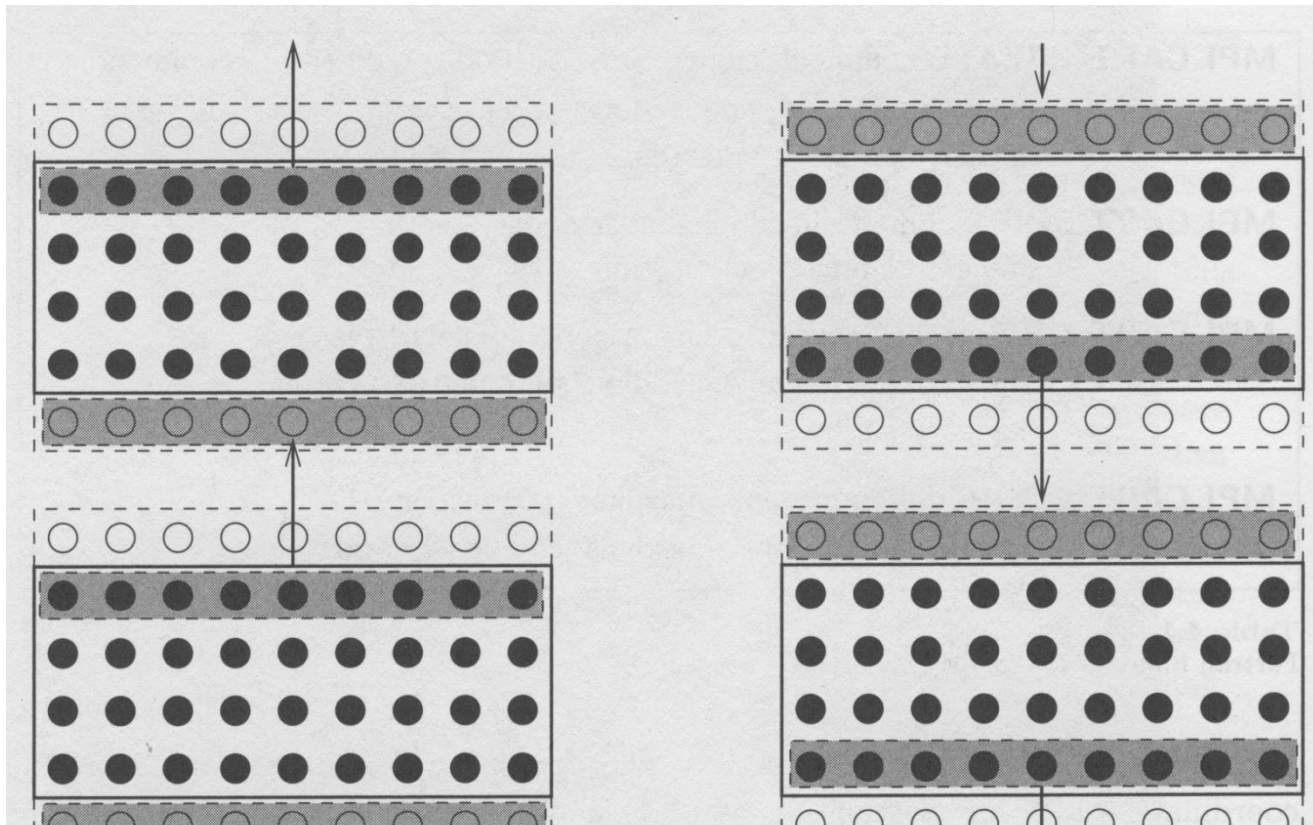
Berechnung
der Poisson Gl.

auf *Star*

Quelle:
Gropp, Lusk & Skjellum
Using MPI



Parallele Bearbeitung einer Matrix II



Austausch
der Randstreifen
im Programmlauf

Quelle:
Gropp, Lusk & Skjellum
Using MPI



Universität Hamburg

DER FORSCHUNG | DER LEHRE | DER BILDUNG



Danke das wars!