Technische Universität München
Institut für Informatik
Prof. Dr. Thomas Huckle
Dipl.-Inf. Christoph Riesinger

WiSe 2015/16

Parallel Numerics

Exercise 9: Domain Decomposition

1 Poisson Equation

Given is the Poisson problem

$$\begin{aligned} -\Delta u &= 1 \qquad u : \Omega = [0,1]^2 \mapsto \mathbb{R}, \\ u|_{\partial\Omega} &= 0 \end{aligned}$$

with homogeneous Dirichlet boundary conditions.

a) Derive the following 5-point stencil for this problem using Finite Differences for equidistant grids:

$$\frac{1}{h^2}\begin{bmatrix} & -1 & \\ -1 & 4 & -1 \\ & -1 & \end{bmatrix}$$

For equidistant and regular grid and with Taylor series we get

$$u(x+h) = u(x) + hu'(x) + \frac{h^2u''(x)}{2!} + \frac{h^3u'''(x)}{3!} + \mathcal{O}(h^4) \tag{1}$$

$$u(x-h) = u(x) - hu'(x) + \frac{h^2u''(x)}{2!} - \frac{h^3u'''(x)}{3!} + \mathcal{O}(h^4) \tag{2}$$

Adding (??) and (??) yields

$$\begin{aligned} u(x+h) + u(x-h) &= 2u(x) + h^2u''(x) + \mathcal{O}(h^4) \\ u''(x) &\approx \frac{u(x+h) - 2u(x) + u(x-h)}{h^2} \end{aligned}$$

Stencil for 1D is $h^{-2}[1 \ -2 \ 1]$. 5-point stencil for 2D ($-\Delta u = -\partial_{xx} - \partial_{yy}$):

$$h^{-2}\begin{bmatrix} 0 & 0 & 0 \\ -1 & 2 & -1 \\ 0 & 0 & 0 \end{bmatrix} + h^{-2}\begin{bmatrix} 0 & -1 & 0 \\ 0 & 2 & 0 \\ 0 & -1 & 0 \end{bmatrix} = h^{-2}\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

b) Set up the resulting matrix A_4 for a 4×4 grid. Enumerate the unknowns in ascending order.

$$A_4 = \begin{pmatrix} 4 & -1 & 0 & 0 & -1 & 0 & & & & \\ -1 & 4 & -1 & 0 & 0 & -1 & 0 & & & \\ 0 & -1 & 4 & -1 & 0 & 0 & -1 & 0 & & \\ 0 & 0 & -1 & 4 & 0 & 0 & 0 & -1 & 0 & \\ -1 & 0 & 0 & 0 & 4 & -1 & 0 & 0 & -1 & \ddots \\ & & & \ddots & & & \ddots & & & \end{pmatrix}$$

c) Set up the resulting matrix A_9 for a 9×9 grid. Enumerate the unknowns in ascending order.
Number of zeros between diagonals is $9 - 2 = 7$.

$$A_9 = \begin{pmatrix} 4 & -1 & 0 & \overset{\#0s:5}{\cdots} & 0 & -1 & 0\ldots & & & \\ -1 & 4 & -1 & 0 & \ldots & 0 & -1 & 0\ldots & & \\ \vdots & & & & & & \vdots & & & \\ -1 & 0 & \overset{\#0s:5}{\cdots} & 0 & -1 & 4 & -1 & 0\ldots 0 & -1 & 0 \\ & & & \ddots & & & & \ddots & & \ddots \end{pmatrix}$$

d) Set up the resulting matrix A_9 for a 9×9 grid. Enumerate the unknowns according to the red-black GS.
A_9 has block structure:

$$A_9 = \begin{pmatrix} 4 & 0 & 0 & 0 & 0 & 0 & \ldots & -1 & 0 & 0 & 0 & -1 & 0 & \ldots & \\ 0 & 4 & 0 & 0 & 0 & 0 & \ldots & -1 & -1 & 0 & 0 & 0 & -1 & 0 & \ldots \\ 0 & 0 & 4 & 0 & 0 & 0 & \ldots & 0 & -1 & -1 & 0 & 0 & 0 & -1 & \ldots \\ 0 & 0 & 0 & 4 & 0 & 0 & \ldots & & & & & \ddots & & \ldots & \\ & & & \ddots & & & & & & & & \ddots & & & \end{pmatrix} = \left(\begin{array}{c|c} 4I & F \\ \hline F^T & 4I \end{array}\right)$$

2 Domain Decomposition I: Dissection

a) Reenumerate the unknowns of the 9×9 grid: First enumerate the unknowns within $\Omega_1 = [1..4] \times [1..4]$, then the unknowns within $\Omega_2 = [1..4] \times [6..9]$, $\Omega_3 = [6..9] \times [1..4]$ and $\Omega_4 = [6..9] \times [6..9]$. Finally, assign a number to all the unknowns left (Ω_5).

The enumeration can be written as:

1	2	3	4	69	17	18	19	20
5	6	7	8	70	21	22	23	24
9	10	11	12	71	25	26	27	28
13	14	15	16	72	29	30	31	32
65	66	67	68	81	73	74	75	76
33	34	35	36	77	49	50	51	52
37	38	39	40	78	53	54	55	56
41	42	43	44	79	57	58	59	60
45	46	47	48	80	61	62	63	64

b) Set up the global stiffness matrix $\hat{A}_9$ according to the new enumeration. Write $\hat{A}_9$ using the matrix A_4.

$$\hat{A}_9 = \left(\begin{array}{cccc|c} A_4 & 0 & 0 & 0 & F_1 \\ 0 & A_4 & 0 & 0 & F_2 \\ 0 & 0 & A_4 & 0 & F_3 \\ 0 & 0 & 0 & A_4 & F_4 \\ \hline F_1^T & F_2^T & F_3^T & F_4^T & F_5 \end{array}\right)$$

c) Assume all unknowns except the unknowns within Ω_1 are known. How does the equation system for Ω_1 look like?
We receive a small system compared to the 9×9-system:

$$A_4 u_{\Omega_1} = -F_1 u_{\Omega_5} + rhs$$

d) Assume all unknowns within $\Omega_1, \Omega_2, \Omega_3$ and Ω_4 are known. How does the equation system for Ω_5 look like?

$$\underbrace{F_1^T u_{\Omega_1} + F_2^T u_{\Omega_2} + F_3^T u_{\Omega_3} + F_4^T u_{\Omega_4}}_{known} + F_5 u_{\Omega_5} = rhs$$

e) Define a parallel algorithm in pseudocode (own words) using the two small equation systems derived before.

```
Set all u-values to 0
while iterate
      Step #1:  u_Ω5 remains
          solve/update iteratively u_Ω1, u_Ω2, u_Ω3, u_Ω4 in parallel
      Step #2:  solve/update u_Ω5 with given u-values everywhere else
```

The small interface domain is solved sequentially, whereas the huge disconnected subdomains are solved in parallel.

3 Domain Decomposition II: A Non-overlapping Method

To set up matrix A, one has to set up one equation per unknown. Instead of traversing all the unknowns of the grid, one could also examine every cell and accumulate the whole system using the element-wise operator

$$\frac{1}{h^2}\begin{pmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} & 0 \\ -\frac{1}{2} & 1 & 0 & -\frac{1}{2} \\ -\frac{1}{2} & 0 & 1 & -\frac{1}{2} \\ 0 & -\frac{1}{2} & -\frac{1}{2} & 1 \end{pmatrix}$$

a) Assume the unknowns within the 9×9 grid are enumerated in ascending order. How is the 14th line of matrix A_9 set up step by step?
Considering the element-wise and vertex-wise assembly of the matrix with

$$\begin{array}{c|c} I & II \\ \hline III & IV \end{array}$$

yields following steps for 14th line:

$$\text{After step I:} \quad \overbrace{0 \ \dots \ 0}^{\#0s:4} \ -0.5 \ \overbrace{0 \ \dots \ 0}^{\#0s:7} \ -0.5 \ 1 \ 0 \ \dots$$

$$\text{After step II:} \quad \overbrace{0 \ \dots \ 0}^{\#0s:4} \ -1 \ \overbrace{0 \ \dots \ 0}^{\#0s:7} \ -0.5 \ 2 \ -0.5 \ 0 \ \dots$$

$$\text{After step III:} \quad \overbrace{0 \ \dots \ 0}^{\#0s:4} \ -1 \ \overbrace{0 \ \dots \ 0}^{\#0s:7} \ -1 \ 3 \ -0.5 \ \overbrace{0 \ \dots \ 0}^{\#0s:7} \ -0.5 \ 0 \ \dots$$

$$\text{After step IV:} \quad \overbrace{0 \ \dots \ 0}^{\#0s:4} \ -1 \ \overbrace{0 \ \dots \ 0}^{\#0s:7} \ -1 \ 4 \ -1 \ \overbrace{0 \ \dots \ 0}^{\#0s:7} \ -1 \ 0 \ \dots$$

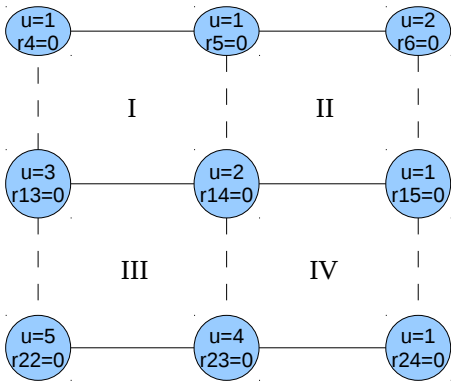
b) Give a Jacobi-wise processing scheme using the element-wise matrices. Focus on a subsection of the whole grid (e.g. scheme for vertex 14) and use the residual formulation. Note, that there is no formula requested, just a processing scheme.
With known residual, a vertex can be updated via

$$u = u + \frac{1}{4}r. \qquad (Jacobi - scheme \rightarrow D^{-1} = diag\left(\frac{1}{4}, \dots, \frac{1}{4}\right))$$

The residual can be computed element-wise by traversing the domains

$$r = b - Au = b - A_I u - A_{II} u - A_{III} u - A_{IV} u,$$

where $I \rightarrow II \rightarrow III \rightarrow IV$ are the regions processed succesively. In the following we compute Au for all regions and denote the result as r. In each domain the residual is updated. Example considering A_9:



We enter element I and update all four adjacent residuals. Using the element-wise operator we receive the residuals:

$$
\begin{aligned}
r_4 &= 1 \cdot 1 - \frac{1}{2} \cdot 3 - \frac{1}{2} \cdot 1 = -1 \\
r_5 &= 1 \cdot 1 - \frac{1}{2} \cdot 1 - \frac{1}{2} \cdot 2 = -\frac{1}{2} \\
r_{13} &= 1 \cdot 3 - \frac{1}{2} \cdot 1 - \frac{1}{2} \cdot 2 = \frac{3}{2} \\
r_{14} &= 1 \cdot 2 - \frac{1}{2} \cdot 3 - \frac{1}{2} \cdot 1 = 0
\end{aligned}
$$

We now enter element II and update all four adjacent residuals using the computed values from element I via

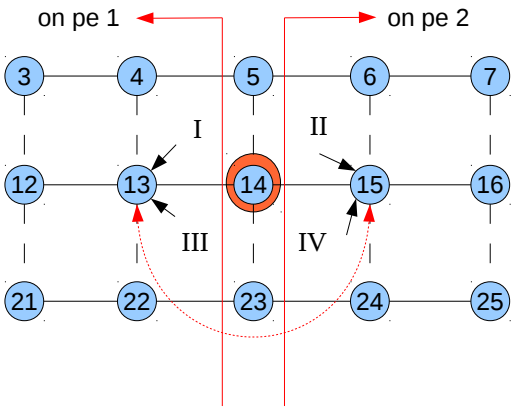
$$
\begin{aligned}
r_5 &= \underbrace{-\frac{1}{2}}_{from\ element\ I} + 1 \cdot 1 - \frac{1}{2} \cdot 2 - \frac{1}{2} \cdot 2 = -\frac{3}{2} \\
r_{14} &= \underbrace{0}_{from\ element\ I} + 1 \cdot 2 - \frac{1}{2} \cdot 1 - \frac{1}{2} \cdot 1 = 1 \\
&\dots
\end{aligned}
$$

After element IV was processed, vertex 14 must take the right-hand side into account via

$$
r_{14} = b_{14} - r_{14}. \tag{3}
$$

Note that only by triggering (**??**) the residual contains the correct value. Finally, we can update u as observed above. The result is a Jacobi scheme, as updates do not influence each other immediately. The global stiffness matrix is never set up (saves lots of memory).

c) Assume the grid is split up into two sub domains $[1..5] \times [1..9]$ and $[5..9] \times [1..9]$. Derive a parallel Jacobi-wise processing scheme without setting up the stiffness matrix explicitly.
We consider an example with 2 domains:



In words:

- Split up domain's elements among the individual processors (here 2 partitions). The vertices along the splitting belong to both processors, i.e. both processors hold their value.
- Compute residuals element-wise, i.e. run over local grid (see b)).
- For local vertices the solution can be updated locally without problems (see b)).
- Problems occur for vertices along the splitting (consider vertex 14 on the boundary):
  - Left node computes contributions from I and III.
  - Right node computes contributions from II and IV.
  - Both exchange their residuals. Communication among the two processors is necessary.
  - Both update the value of vertex 14 on the boundary, i.e. the duplicated value is consistent again.

4) The Quality of a Partition

From a parallelisation point of view, a partition is "optimal" if it has a small boundary, a big volume and few neighbouring partitions.

a) Give reasons for this statement.
Little communication. Small amount of communication with other processors. Most of the computation/work can be performed locally.

b) What would be an optimal partition for the 9×9 grid.
Optimal partitioning would be a spherical partitioning, because a sphere has minimal surface-to-volume ratio. As spherical partitioning is not practical another (less optimal) possibility is to use space-filling curves, kd-trees, or octal trees.

c) Outlook: In the lecture "Algorithmen des Wissenschaftlichen Rechnens" it is proven,

that space–filling curves are Hölder continuous. Assume one splits up a computational domain according to the approximating polygon of a space–filling curve. What does this mean for the quality of the partition?
It can be proven that the splitting according to the approximating polygon of e.g. the Hilbert curve provides partitions with a surface-to-volume ratio similar to the one of a sphere up to a constant factor. Therefore, the partitions provide the advantages of (i).