Scientific Computing - Exercise Sheet 1

Jonathan Hellwig, Jule Schütt, Mika Tode, Giuliano Taccogna

May 7, 2021

1 Exercise

(a) Noch nichts gemacht:

Input: $\mathbf{A} \in \mathbb{R}^{n \times n} \quad b, x_0 \in \mathbb{R}^n$

$h_0 = \mathbf{A}x_0$
$r_0 = b - h_0$
$p_0 = r_0$
$\beta_0 = r_0^T \cdot r_0$
for $k = 1, 2, \ldots$ **do**
  $h_{k-1} = \mathbf{A}p_{k-1}$ (matrix multiplication)
  $\gamma_{k-1} = p_{k-1}^T \cdot h_{k-1}$ (1. loop)
  $\alpha_{k-1} = \frac{\beta^{k-1}}{\gamma^{k-1}}$
  $x_k = x_{k-1} + \alpha_{k-1}p_{k-1}$ (2 loop)
  $r_k = r_{k-1} - \alpha_{k-1}h_{k-1}$ (3. loop)
  $\beta_k = r_k^T \cdot r_k$ (4. loop)
  $p_k = r_k + \frac{\beta_k}{\beta_{k-1}}p_{k-1}$ (5. loop)
end for

(b) The idea is to split the vectors such that the first coordinates were computed by one processor and the last coordinates were computed by the second processor. Therefore i denotes the counter, which is needed for the loop operation. Notice that for the Vectormultiplications in the loops 1 and 4 normally also a **reduce** operation should be included.

Input: $\mathbf{A} \in \mathbb{R}^{n \times n} \quad b, x_0 \in \mathbb{R}^n$

$h_0 = \mathbf{A}x_0$
begin parallel private(i, r_0) **shared**(b, h_0)
$r_0 = b - h_0$
end parallel
$p_0 = r_0$
begin parallel private(i, β_0) **shared**(r_0) **end parallel**
$\beta_0 = r_0^T \cdot r_0$
for $k = 1, 2, \ldots$ **do**
  $h_{k-1} = \mathbf{A}p_{k-1}$ (matrix multiplication)

begin parallel private(i, γ_{k-1}) **shared**(p_{k-1}, h_{k-1})
$\gamma_{k-1} = p_{k-1}^T \cdot h_{k-1}$ (1. loop)
end parallel
$\alpha_{k-1} = \frac{\beta^{k-1}}{\gamma^{k-1}}$
begin parallel private(i, x_k, p_k) **shared**$(r_{k-1}, r_k, x_{k-1}, h_{k-1}, \alpha_{k-1}, \beta_k, \beta_{k-1}, p_{k-1})$
$x_k = x_{k-1} + \alpha_{k-1} p_{k-1}$ (2 loop)
$r_k = r_{k-1} - \alpha_{k-1} h_{k-1}$ (3. loop)
$\beta_k = r_k^T \cdot r_k$ (4. loop)
$p_k = r_k + \frac{\beta_k}{\beta_{k-1}} p_{k-1}$ (5. loop)
end parallel
end for

(c) We determine that for all parallelizable parts Processor 0 works on the for-loop for $i = 1, ..., n/2$ and Processor 1 works on the for-loop for $i = n/2+1, ..., n$, with $n/2$ possibly rounded.
Accordingly $v^{(j)}$ denotes a scalarproduct or a vector which is calculated by processor $j = 0, 1$. If $v^{(j)}$ denotes a Vector we mean $v^{(0)} = (v(1), v(2), ..., v(n/2))^T$ and $v^{(1)} = (v(n/2+1), v(n/2+2), ..., v(n))^T$. If $v^{(j)}$ denotes a scalarproduct, we mean $v^{(j)} = H^{T(j)} \cdot K^{(j)}$ for $H, K \in \mathbb{R}^n$.
If a processor recives data through the **recv** command, we implicitly include in **recv** that the recieved data is combined with the data that was calculated on the processor itself in the right way.
That means if one half of a vector $v^{(i)}$ is recieved **recv** creates the whole vector $v = v^{(0)} + v^{(1)}$. If "one half" of a scalarproduct is recived **recv** adds the corresponding scalarproducts to the final scalarproduct. If the processor is not specified in a calculation, the calculation is executed on both processors.
For the first processor (index 0):
Input: $\mathbf{A} \in \mathbb{R}^{n \times n}$ $b, x_0 \in \mathbb{R}^n$

$h_0 = \mathbf{A} x_0$
$r_0^{(0)} = b^{(0)} - h_0^{(0)}$
barrier
send$(r_0^{(0)})$
recv$(r_0^{(1)})$
$p_0 = r_0$
$\beta_0^{(0)} = r_0^{(0)T} \cdot r_0^{(0)}$
barrier
send$(\beta_0^{(0)})$
recv$(\beta_0^{(1)})$
for $k = 1, 2, \ldots$ **do**
$h_{k-1} = \mathbf{A} p_{k-1}$ (matrix multiplication)
$\gamma_{k-1}^{(0)} = (p_{k-1}^{(0)})^T \cdot h_{k-1}^{(0)}$ (1. loop)
barrier
send$(\gamma_{k-1}^{(0)})$

recv$(\gamma_{k-1}^{(1)})$

$\alpha_{k-1} = \frac{\beta^{k-1}}{\gamma^{k-1}}$

$x_k^{(0)} = x_{k-1}^{(0)} + \alpha_{k-1}^{(0)} p_{k-1}^{(0)}$ (2 loop)

$r_k^{(0)} = r_{k-1}^{(0)} - \alpha_{k-1}^{(0)} h_{k-1}^{(0)}$ (3. loop)

$\beta_k^{(0)} = (r_k^{(0)})^T \cdot r_k^{(0)}$ (4. loop)

barrier

send$(\beta_k^{(0)})$

recv$(\beta_k^{(1)})$

$p_k^{(0)} = r_k^{(0)} + \frac{\beta_k^{(0)}}{\beta_{k-1}^{(0)}} p_{k-1}$ (5. loop)

barrier

send$(p_k^{(0)})$

recv$(p_k^{(1)})$

end for

(d)