HLR - Hochleistungsrechnen

Aufgabenblatt 4

Merle Hoffmann, Joël Miramon, Max Press

1. Datenaufteilung:
In vielen Fällen muss die Datenaufteilung explizit und manuell vorgenommen werden. Überlegen Sie sich eine geeignete effiziente Aufteilung der Matrix auf die Threads.

1. Entwickeln Sie eine generische Formel für i(Interlines) $\in \mathbb{N}_0$ und t(Threads) $\in \mathbb{N}$. Beachten Sie, wie die Matrixgröße mit i aufgeschlüsselt wird - achten Sie vor allem auf die Randzeilen-/spalten.

   Die Randzeilen und - spalten werden für die Berrechnung nicht benötigt, weshalb wir nur die inneren Elemente auf die Threads aufteilen. Wir teilen die Matrix elementweise, um eine möglichst gleichmäßige Aufteilung zu bekommen. Dafür teilen wir die Anzahl der inneren Elemente in der Matrix durch die Anzahl der Threads. Falls die Elemente nicht gleichmäßig auf alle Threads aufgeteilt werden können, also ein Rest bei der Division entsteht, wird dieser Rest auf die Threads verteilt. Somit unterscheidet sich die Chunksize der Threads um maximal ein Element.

Interlines	$i \in \mathbb{N}_0$
Threads	$t \in \mathbb{N}$
Aktueller Thread	T
Chunksize für Thread T	$C_T(i,t)$

   $$C_T(i,t) = \begin{cases} \lfloor (8*i+9-2)^2/t \rfloor + 1, & \text{falls } T \leq ((8*i+9-2)^2 \bmod t) \\ \lfloor (8*i+9-2)^2/t \rfloor, & \text{sonst.} \end{cases}$$

2. Schreiben Sie im Pseudocode auf, wie Sie gemäß dieser Formel die Matrix auf die Threads aufteilen und berechnen würden. Beschränken Sie sich auf die zwei for-Schleifen, wobei die Indizes i und j explizit in den Schleifen und beim Zugriff auf die Matrix angegeben werden müssen.

```
// Definiere struct für Thread Parameter mit offset, chunksize und
// allen weiteren Parametern für Berechnung, zB: Matrix_In, Matrix_Out
void calculate:
  ⋮
  int N ← 8 * i + 9 − 2
  int inner_elements ← N * N
  int intervall ← inner_elements/t
  int rest ← inner_elements mod t
  while term_iteration > 0 do
    ⋮
    for T = 0; T < t; T++ do
      // Allokiere und initialisiere Thread Parameter
      if T < rest then
        int offset ← T * (intervall + 1)
        int chunksize ← intervall + 1
      else
        int offset ← T * intervall + rest
        int chunksize ← intervall
      end if
      // Erzeuge einen Thread
    end for
  end while
void* thread_function:
  ⋮
  int offset_end ← offset + chunksize − 1
  int i_start ← offset mod N + 1
  int j_start ← offset/N + 1
  int i_end ← offset_end mod N + 1
  int j_end ← offset_end/N + 1
  for i = i_start; i ≤ i_end; i++ do
    for j = 1; j < N; j++ do
      if i == i_start && j < j_start then    ▷ Ignoriere Elemente vor offset
        continue
      else if i == i_end && j > j_end then    ▷ Breche nach offset_end ab
        break
      end if
      star = 0.25 * (Matrix_In[i − 1][j] + Matrix_In[i][j − 1]
                  + Matrix_In[i][j + 1] + Matrix_In[i + 1][j])
      ⋮
      Matrix_Out[i][j] = star
    end for
  end for
```

3. Visualisieren Sie Ihre Aufteilung (grafisch). Zeichnen Sie jeweils eine Matrix für i=0, t=4 und i=1, t=4. Machen Sie dabei auch die Randzeilen, die nicht Teil der Berechnung sind, kenntlich.

   (a) $i = 0, t = 4$

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

   Thread T = 1:

$$C_1(0,4) = \lfloor((8*0+9-2)^2/4)+1\rfloor = \lfloor 12.25\rfloor + 1 = 13$$

   Thread T = 2: Thread T = 3: Thread T = 4:

$$C_{2,3,4}(0,4) = \lfloor(8*0+9-2)^2/4\rfloor = \lfloor 12.25\rfloor = 12$$

   (b) $i = 1, t = 4$

	00	01	**02**	03	**04**	05	**06**	07	**08**	09	**10**	11	**12**	13	**14**	15	**16**
00	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
01	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
02	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
03	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
04	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
05	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
06	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
07	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
08	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
09	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
10	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
11	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
12	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
13	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
14	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
15	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
16	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

   Thread T = 1:

$$C_1(1,4) = \lfloor((8*1+9-2)^2/4)+1\rfloor = \lfloor 56.25\rfloor + 1 = 57$$

   Thread T = 2: Thread T = 3: Thread T = 4:

$$C_{2,3,4}(1,4) = \lfloor(8*1+9-2)^2/4\rfloor = \lfloor 56.25\rfloor = 56$$