Implementing a systolic algorithm for QR factorization on multicore clusters with PaRSEC

Guillaume Aupy, Mathieu Faverge, Yves Robert,
Jakub Kurzak, Piotr Luszczek, Jack Dongarra

August 27, 2013

PROPER 2013

Motivation, 3D-Torus

Current and upcoming HPC systems have a 3D-Torus underlying interconnection topology.

- Blue Gene/L is a 3D torus of size $64 \times 32 \times 32$,
- Kraken, a Cray XT 5, is a 3D torus of size $25 \times 16 \times 24$,
- Cray XT3 and XT4 also are architectures based on a 3D torus.

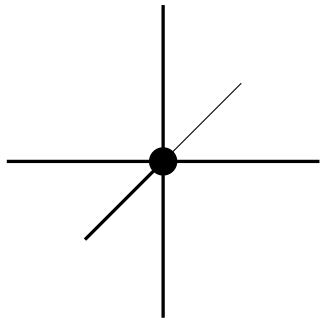


Figure : A node of a 3D-torus and its direct communication links

Motivation, existing QR algorithm

Most recent works either

- Do not take into account the architecture
- Consider a 1D-torus (2/3 of the communication links are not used in an optimal fashion)
- Consider a 2D-torus (1/3 of the communication links are not used in an optimal fashion)

With the advent of exascale it becomes necessary to take the architecture into account.

Motivation, existing QR algorithm

Most recent works either

- Do not take into account the architecture
- Consider a 1D-torus (2/3 of the communication links are not used in an optimal fashion)
- Consider a 2D-torus (1/3 of the communication links are not used in an optimal fashion)

With the advent of exascale it becomes necessary to take the architecture into account.

Introduction

QR factorization of a matrix is a decomposition of a matrix A into a product A=QR

- Q is an orthogonal matrix
- R is an upper triangular matrix.

QR factorization is often used to solve the linear least squares problem, and is the basis for a particular eigenvalue algorithm, the QR algorithm.

Wikipedia

Generic QR algorithm.

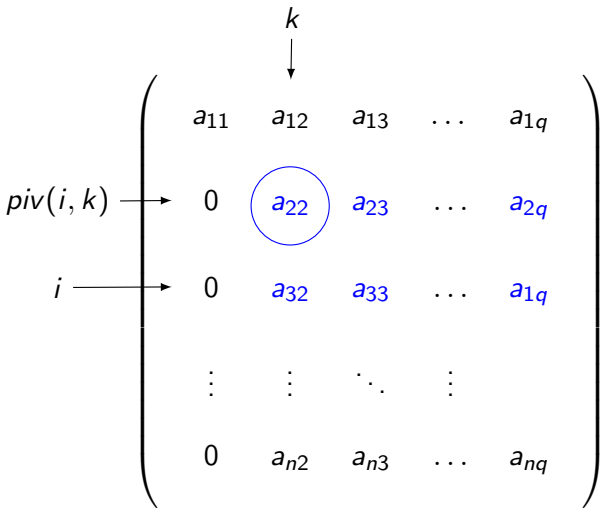
begin
  for k $= 0$ *to* $\min(m, n) - 1$ **do**
    for i $= k + 1$ *to* $m - 1$ **do**
      $elim(i, \textbf{CurPiv}(i, k), k)$
    end
  end
end

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1q} \\ 0 & a_{22} & a_{23} & \dots & a_{2q} \\ 0 & a_{32} & a_{33} & \dots & a_{1q} \\ \vdots & \vdots & \ddots & \vdots & \\ 0 & a_{n2} & a_{n3} & \dots & a_{nq} \end{pmatrix}$$

- k: panel index
- **orthogonal transformation** to zero out tile (i, k)

Generic QR algorithm.

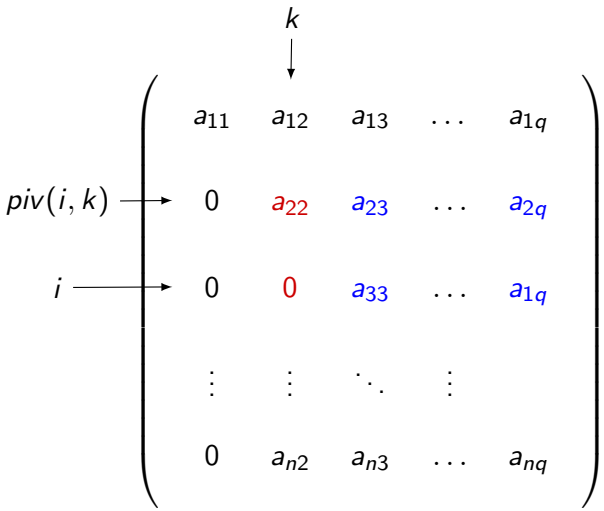
begin
  for k = 0 *to* $\min(m, n) - 1$ **do**
    for i = $k + 1$ *to* $m - 1$ **do**
      $elim(i, \textbf{CurPiv}(i, k), k)$
    end
  end
end



- k: panel index
- **orthogonal transformation** to zero out tile (i, k)

Generic QR algorithm.

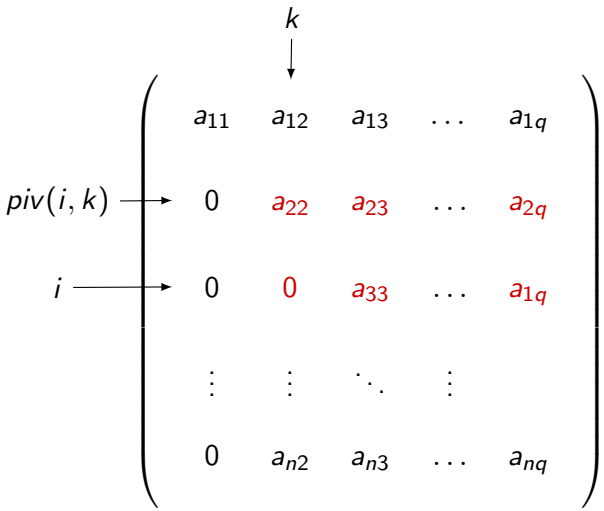
begin
  for k = 0 *to* $\min(m, n) - 1$ **do**
    for i = $k + 1$ *to* $m - 1$ **do**
      $elim(i, \textbf{CurPiv}(i, k), k)$
    end
  end
end



- k: panel index
- **orthogonal transformation** to zero out tile (i, k)

Generic QR algorithm.

begin
  for k = 0 *to* $\min(m, n) - 1$ **do**
    for i = $k + 1$ *to* $m - 1$ **do**
      $elim(i, \textbf{CurPiv}(i, k), k)$
    end
  end
end



- k: panel index
- **orthogonal transformation** to zero out tile (i, k)

Generic QR algorithm.

begin
  for k $= 0$ *to* $\min(m, n) - 1$ **do**
    for i $= k + 1$ *to* $m - 1$ **do**
      $elim(i, \textbf{CurPiv}(i, k), k)$
    end
  end
end

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1q} \\ 0 & a_{22} & a_{23} & \dots & a_{2q} \\ 0 & 0 & a_{33} & \dots & a_{1q} \\ \vdots & \vdots & \ddots & \vdots & \\ 0 & a_{n2} & a_{n3} & \dots & a_{nq} \end{pmatrix}$$

- k: panel index
- **orthogonal transformation** to zero out tile (i, k)

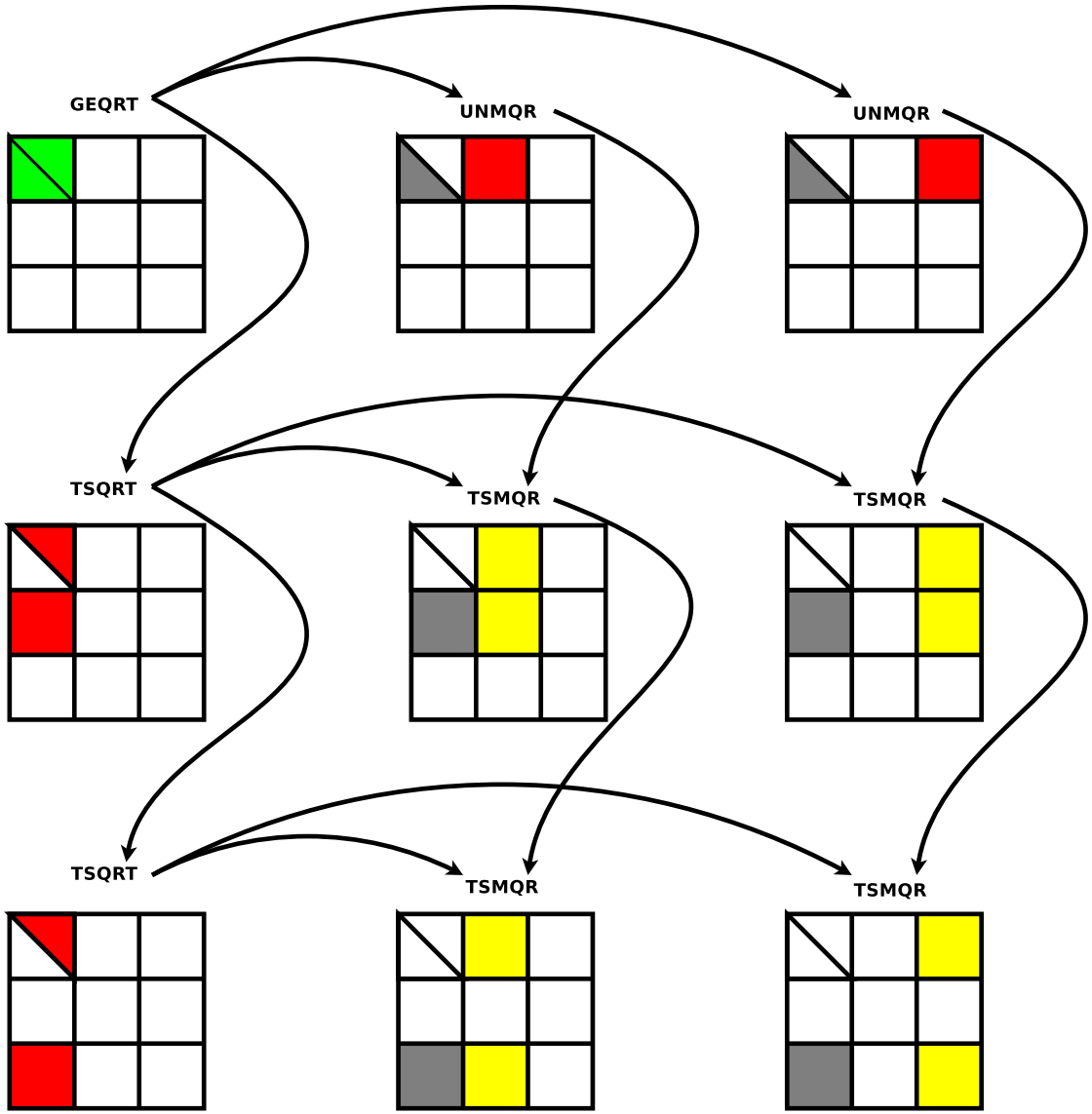
Generic QR algorithm.

begin
  for k $= 0$ *to* $\min(m, n) - 1$ **do**
    for i $= k + 1$ *to* $m - 1$ **do**
      $elim(i, \textbf{CurPiv}(i, k), k)$
    end
  end
end

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1q} \\ 0 & a_{22} & a_{23} & \dots & a_{2q} \\ 0 & 0 & a_{33} & \dots & a_{1q} \\ \vdots & \vdots & \ddots & \vdots & \\ 0 & 0 & a_{n3} & \dots & a_{nq} \end{pmatrix}$$

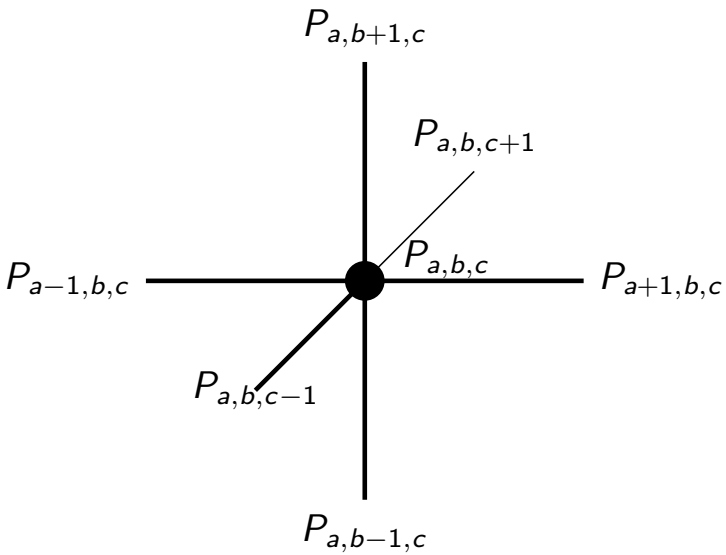
- k: panel index
- **orthogonal transformation** to zero out tile (i, k)

Tile QR factorization

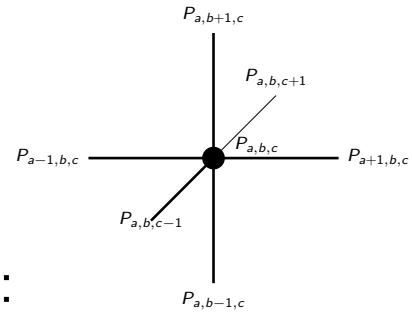


DAG for one step of QR factorization

Within a $p \times q \times r$ 3D torus,
processor $P_{a,b,c}$ has a direct communication link with 6 processors:



- $P_{a-1 \mod p,b,c}$
- $P_{a+1 \mod p,b,c}$
- $P_{a,b-1 \mod q,c}$
- $P_{a,b+1 \mod q,c}$
- $P_{a,b,c-1 \mod r}$
- $P_{a,b,c+1 \mod r}$



$m \times n$ tile matrix. Tiles are mapped as follows:

- we use a two-level cyclic distribution for the rows (directions a and b in the torus)
- a cyclic distribution for the columns (direction c in the torus).

Formally as follows: proc P_{abc} is assigned all tiles $T_{t,s}$ such that

- $t \equiv b \mod q$, $\frac{t-b}{q} \equiv a \mod p$
- $s \equiv c \mod r$

- 2-level-cyclic distribution for the rows
- cyclic distribution for the columns

$P_{0,0,0}$ $P_{1,0,0}$ $P_{2,0,0}$

$P_{0,1,0}$ $P_{1,1,0}$ $P_{2,1,0}$

$P_{0,2,0}$ $P_{1,2,0}$ $P_{2,2,0}$

0 × ×
1 × ×
2 × ×
3 × ×
4 × ×
5 × ×
6 × ×
7 × ×
8 × ×
9 × ×
10 × ×
11 × ×
12 × ×
13 × ×
14 × ×
15 × ×
16 × ×
17 × ×
18 × ×
19 × ×
20 × ×
21 × ×
22 × ×
23 × ×
24 × ×
25 × ×
26 × ×

- 2-level-cyclic distribution for the rows
- cyclic distribution for the columns

0

$P_{0,0,0}$ $P_{1,0,0}$ $P_{2,0,0}$

$P_{0,1,0}$ $P_{1,1,0}$ $P_{2,1,0}$

$P_{0,2,0}$ $P_{1,2,0}$ $P_{2,2,0}$

0 × ×
1 × ×
2 × ×
3 × ×
4 × ×
5 × ×
6 × ×
7 × ×
8 × ×
9 × ×
10 × ×
11 × ×
12 × ×
13 × ×
14 × ×
15 × ×
16 × ×
17 × ×
18 × ×
19 × ×
20 × ×
21 × ×
22 × ×
23 × ×
24 × ×
25 × ×
26 × ×

- 2-level-cyclic distribution for the rows
- cyclic distribution for the columns

0

$P_{0,0,0}$ $P_{1,0,0}$ $P_{2,0,0}$

1

$P_{0,1,0}$ $P_{1,1,0}$ $P_{2,1,0}$

$P_{0,2,0}$ $P_{1,2,0}$ $P_{2,2,0}$

0 × ×
1 × ×
2 × ×
3 × ×
4 × ×
5 × ×
6 × ×
7 × ×
8 × ×
9 × ×
10 × ×
11 × ×
12 × ×
13 × ×
14 × ×
15 × ×
16 × ×
17 × ×
18 × ×
19 × ×
20 × ×
21 × ×
22 × ×
23 × ×
24 × ×
25 × ×
26 × ×

- 2-level-cyclic distribution for the rows
- cyclic distribution for the columns

0

$P_{0,0,0}$ $P_{1,0,0}$ $P_{2,0,0}$

1

$P_{0,1,0}$ $P_{1,1,0}$ $P_{2,1,0}$

2

$P_{0,2,0}$ $P_{1,2,0}$ $P_{2,2,0}$

0 × ×
1 × ×
2 × ×
3 × ×
4 × ×
5 × ×
6 × ×
7 × ×
8 × ×
9 × ×
10 × ×
11 × ×
12 × ×
13 × ×
14 × ×
15 × ×
16 × ×
17 × ×
18 × ×
19 × ×
20 × ×
21 × ×
22 × ×
23 × ×
24 × ×
25 × ×
26 × ×

- 2-level-cyclic distribution for the rows
- cyclic distribution for the columns

0 **3**

$P_{0,0,0}$ $P_{1,0,0}$ $P_{2,0,0}$

1

$P_{0,1,0}$ $P_{1,1,0}$ $P_{2,1,0}$

2

$P_{0,2,0}$ $P_{1,2,0}$ $P_{2,2,0}$

0 × ×
1 × ×
2 × ×
3 × ×
4 × ×
5 × ×
6 × ×
7 × ×
8 × ×
9 × ×
10 × ×
11 × ×
12 × ×
13 × ×
14 × ×
15 × ×
16 × ×
17 × ×
18 × ×
19 × ×
20 × ×
21 × ×
22 × ×
23 × ×
24 × ×
25 × ×
26 × ×

- 2-level-cyclic distribution for the rows
- cyclic distribution for the columns

0 **3**

$P_{0,0,0}$ $P_{1,0,0}$ $P_{2,0,0}$

1 **4**

$P_{0,1,0}$ $P_{1,1,0}$ $P_{2,1,0}$

2

$P_{0,2,0}$ $P_{1,2,0}$ $P_{2,2,0}$

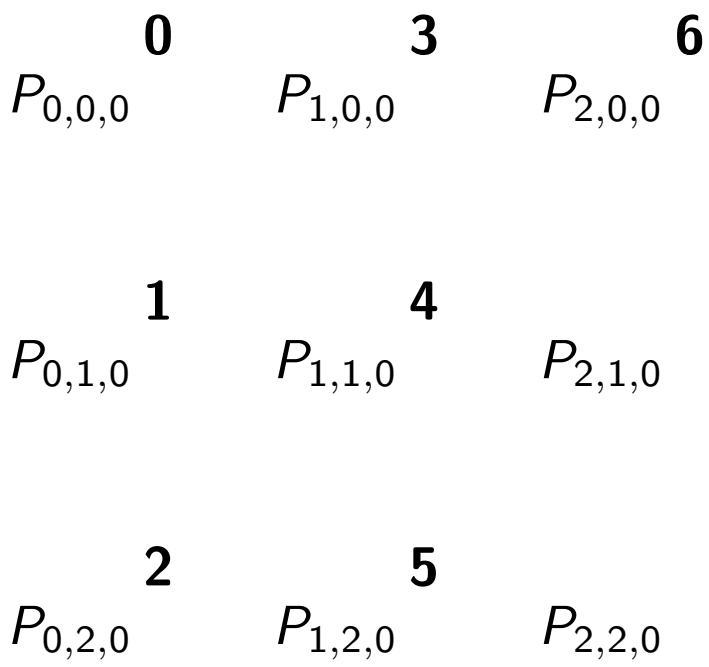
0 × ×
1 × ×
2 × ×
3 × ×
4 × ×
5 × ×
6 × ×
7 × ×
8 × ×
9 × ×
10 × ×
11 × ×
12 × ×
13 × ×
14 × ×
15 × ×
16 × ×
17 × ×
18 × ×
19 × ×
20 × ×
21 × ×
22 × ×
23 × ×
24 × ×
25 × ×
26 × ×

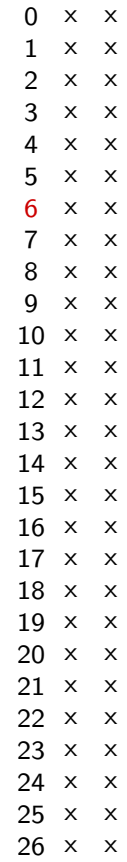
- 2-level-cyclic distribution for the rows
- cyclic distribution for the columns

0	**3**	
$P_{0,0,0}$	$P_{1,0,0}$	$P_{2,0,0}$
1	**4**	
$P_{0,1,0}$	$P_{1,1,0}$	$P_{2,1,0}$
2	**5**	
$P_{0,2,0}$	$P_{1,2,0}$	$P_{2,2,0}$

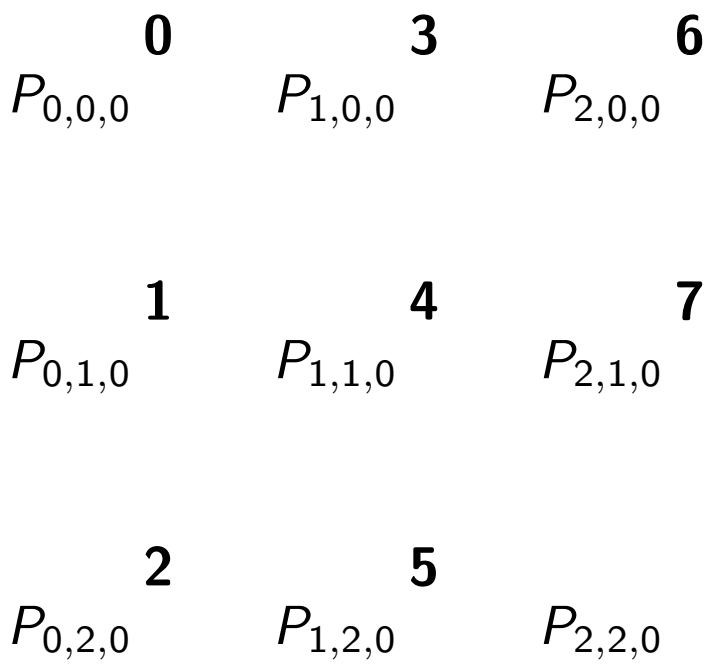
0	×	×
1	×	×
2	×	×
3	×	×
4	×	×
5	×	×
6	×	×
7	×	×
8	×	×
9	×	×
10	×	×
11	×	×
12	×	×
13	×	×
14	×	×
15	×	×
16	×	×
17	×	×
18	×	×
19	×	×
20	×	×
21	×	×
22	×	×
23	×	×
24	×	×
25	×	×
26	×	×

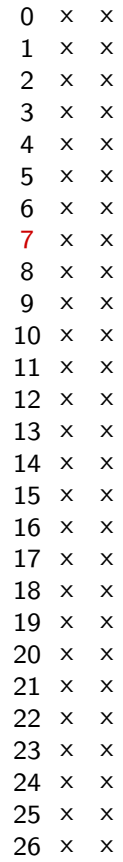
- 2-level-cyclic distribution for the rows
- cyclic distribution for the columns



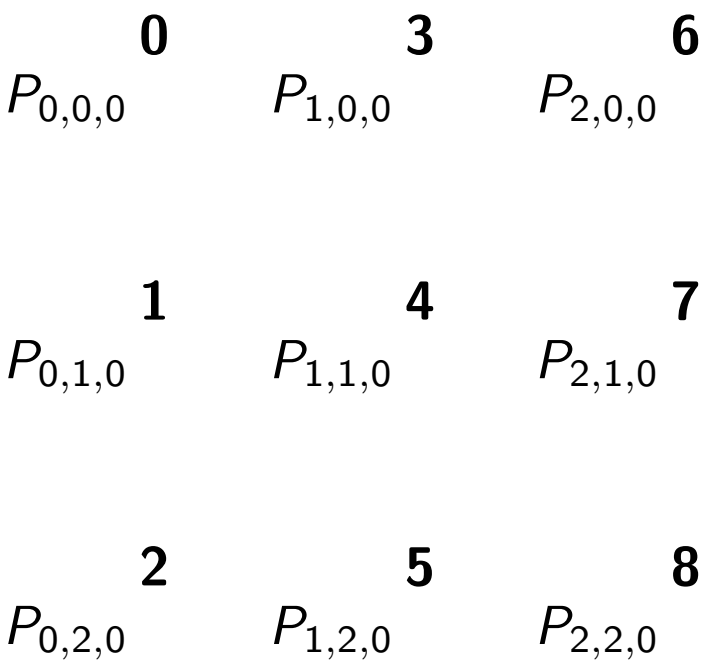


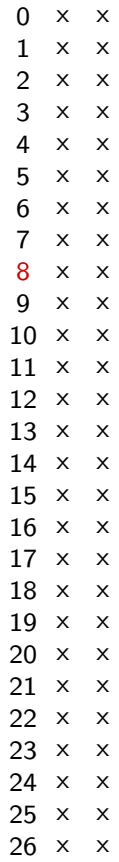
- 2-level-cyclic distribution for the rows
- cyclic distribution for the columns



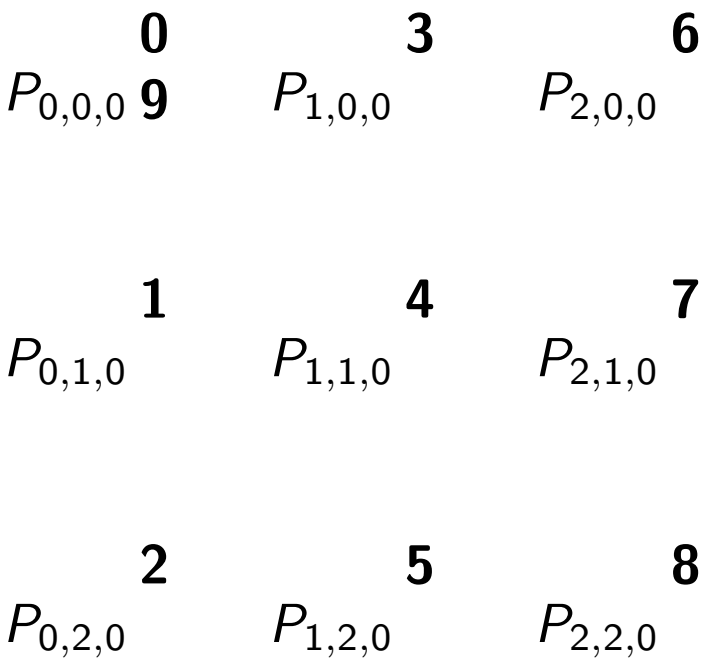


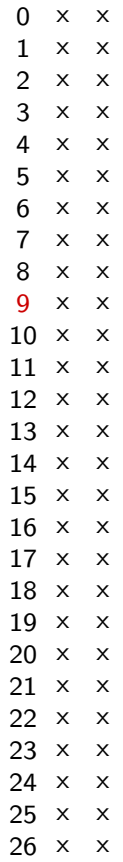
- 2-level-cyclic distribution for the rows
- cyclic distribution for the columns



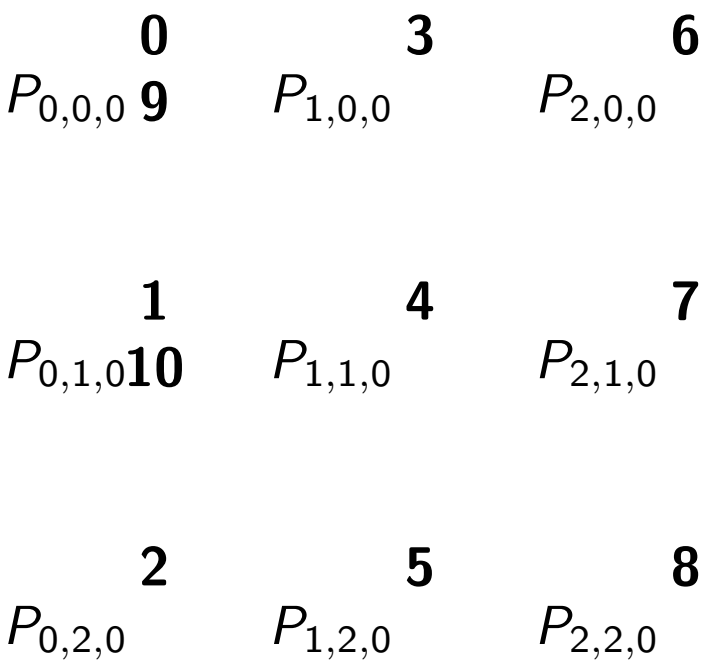


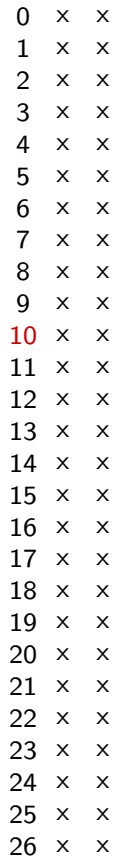
- 2-level-cyclic distribution for the rows
- cyclic distribution for the columns





- 2-level-cyclic distribution for the rows
- cyclic distribution for the columns





- 2-level-cyclic distribution for the rows
- cyclic distribution for the columns

0 $P_{0,0,0}$ **9**	**3** $P_{1,0,0}$	**6** $P_{2,0,0}$
1 $P_{0,1,0}$ **10**	**4** $P_{1,1,0}$	**7** $P_{2,1,0}$
2 $P_{0,2,0}$ **11**	**5** $P_{1,2,0}$	**8** $P_{2,2,0}$

0	×	×
1	×	×
2	×	×
3	×	×
4	×	×
5	×	×
6	×	×
7	×	×
8	×	×
9	×	×
10	×	×
11	×	×
12	×	×
13	×	×
14	×	×
15	×	×
16	×	×
17	×	×
18	×	×
19	×	×
20	×	×
21	×	×
22	×	×
23	×	×
24	×	×
25	×	×
26	×	×

- 2-level-cyclic distribution for the rows
- cyclic distribution for the columns

0 $P_{0,0,0}$ **9**	**3** $P_{1,0,0}$ **12**	**6** $P_{2,0,0}$
1 $P_{0,1,0}$ **10**	**4** $P_{1,1,0}$	**7** $P_{2,1,0}$
2 $P_{0,2,0}$ **11**	**5** $P_{1,2,0}$	**8** $P_{2,2,0}$

0	×	×
1	×	×
2	×	×
3	×	×
4	×	×
5	×	×
6	×	×
7	×	×
8	×	×
9	×	×
10	×	×
11	×	×
12	×	×
13	×	×
14	×	×
15	×	×
16	×	×
17	×	×
18	×	×
19	×	×
20	×	×
21	×	×
22	×	×
23	×	×
24	×	×
25	×	×
26	×	×

- 2-level-cyclic distribution for the rows
- cyclic distribution for the columns

0 $P_{0,0,0}$ **9**	**3** $P_{1,0,0}$ **12**	**6** $P_{2,0,0}$
1 $P_{0,1,0}$ **10**	**4** $P_{1,1,0}$ **13**	**7** $P_{2,1,0}$
2 $P_{0,2,0}$ **11**	**5** $P_{1,2,0}$	**8** $P_{2,2,0}$

0	×	×
1	×	×
2	×	×
3	×	×
4	×	×
5	×	×
6	×	×
7	×	×
8	×	×
9	×	×
10	×	×
11	×	×
12	×	×
13	×	×
14	×	×
15	×	×
16	×	×
17	×	×
18	×	×
19	×	×
20	×	×
21	×	×
22	×	×
23	×	×
24	×	×
25	×	×
26	×	×

- 2-level-cyclic distribution for the rows
- cyclic distribution for the columns

0 $P_{0,0,0}$ **9**	**3** $P_{1,0,0}$ **12**	**6** $P_{2,0,0}$
1 $P_{0,1,0}$ **10**	**4** $P_{1,1,0}$ **13**	**7** $P_{2,1,0}$
2 $P_{0,2,0}$ **11**	**5** $P_{1,2,0}$ **14**	**8** $P_{2,2,0}$

0	×	×
1	×	×
2	×	×
3	×	×
4	×	×
5	×	×
6	×	×
7	×	×
8	×	×
9	×	×
10	×	×
11	×	×
12	×	×
13	×	×
14	×	×
15	×	×
16	×	×
17	×	×
18	×	×
19	×	×
20	×	×
21	×	×
22	×	×
23	×	×
24	×	×
25	×	×
26	×	×

- 2-level-cyclic distribution for the rows
- cyclic distribution for the columns

0 $P_{0,0,0}$ **9**	**3** $P_{1,0,0}$ **12**	**6** $P_{2,0,0}$ **15**
1 $P_{0,1,0}$ **10**	**4** $P_{1,1,0}$ **13**	**7** $P_{2,1,0}$
2 $P_{0,2,0}$ **11**	**5** $P_{1,2,0}$ **14**	**8** $P_{2,2,0}$

0	×	×
1	×	×
2	×	×
3	×	×
4	×	×
5	×	×
6	×	×
7	×	×
8	×	×
9	×	×
10	×	×
11	×	×
12	×	×
13	×	×
14	×	×
15	×	×
16	×	×
17	×	×
18	×	×
19	×	×
20	×	×
21	×	×
22	×	×
23	×	×
24	×	×
25	×	×
26	×	×

- 2-level-cyclic distribution for the rows
- cyclic distribution for the columns

0 $P_{0,0,0}$ **9**	**3** $P_{1,0,0}$ **12**	**6** $P_{2,0,0}$ **15**
1 $P_{0,1,0}$ **10**	**4** $P_{1,1,0}$ **13**	**7** $P_{2,1,0}$ **16**
2 $P_{0,2,0}$ **11**	**5** $P_{1,2,0}$ **14**	**8** $P_{2,2,0}$

0	×	×
1	×	×
2	×	×
3	×	×
4	×	×
5	×	×
6	×	×
7	×	×
8	×	×
9	×	×
10	×	×
11	×	×
12	×	×
13	×	×
14	×	×
15	×	×
16	×	×
17	×	×
18	×	×
19	×	×
20	×	×
21	×	×
22	×	×
23	×	×
24	×	×
25	×	×
26	×	×

- 2-level-cyclic distribution for the rows
- cyclic distribution for the columns

0 $P_{0,0,0}$ **9**	**3** $P_{1,0,0}$ **12**	**6** $P_{2,0,0}$ **15**
1 $P_{0,1,0}$ **10**	**4** $P_{1,1,0}$ **13**	**7** $P_{2,1,0}$ **16**
2 $P_{0,2,0}$ **11**	**5** $P_{1,2,0}$ **14**	**8** $P_{2,2,0}$ **17**

0 × ×
1 × ×
2 × ×
3 × ×
4 × ×
5 × ×
6 × ×
7 × ×
8 × ×
9 × ×
10 × ×
11 × ×
12 × ×
13 × ×
14 × ×
15 × ×
16 × ×
17 × ×
18 × ×
19 × ×
20 × ×
21 × ×
22 × ×
23 × ×
24 × ×
25 × ×
26 × ×

- 2-level-cyclic distribution for the rows
- cyclic distribution for the columns

0 $P_{0,0,0}$ **9** **18**	**3** $P_{1,0,0}$ **12**	**6** $P_{2,0,0}$ **15**
1 $P_{0,1,0}$ **10**	**4** $P_{1,1,0}$ **13**	**7** $P_{2,1,0}$ **16**
2 $P_{0,2,0}$ **11**	**5** $P_{1,2,0}$ **14**	**8** $P_{2,2,0}$ **17**

0	×	×
1	×	×
2	×	×
3	×	×
4	×	×
5	×	×
6	×	×
7	×	×
8	×	×
9	×	×
10	×	×
11	×	×
12	×	×
13	×	×
14	×	×
15	×	×
16	×	×
17	×	×
18	×	×
19	×	×
20	×	×
21	×	×
22	×	×
23	×	×
24	×	×
25	×	×
26	×	×

- 2-level-cyclic distribution for the rows
- cyclic distribution for the columns

0 $P_{0,0,0}$ **9** **18**	**3** $P_{1,0,0}$ **12**	**6** $P_{2,0,0}$ **15**
1 $P_{0,1,0}$ **10** **19**	**4** $P_{1,1,0}$ **13**	**7** $P_{2,1,0}$ **16**
2 $P_{0,2,0}$ **11**	**5** $P_{1,2,0}$ **14**	**8** $P_{2,2,0}$ **17**

0 × ×
1 × ×
2 × ×
3 × ×
4 × ×
5 × ×
6 × ×
7 × ×
8 × ×
9 × ×
10 × ×
11 × ×
12 × ×
13 × ×
14 × ×
15 × ×
16 × ×
17 × ×
18 × ×
19 × ×
20 × ×
21 × ×
22 × ×
23 × ×
24 × ×
25 × ×
26 × ×

- 2-level-cyclic distribution for the rows
- cyclic distribution for the columns

0 $P_{0,0,0}$ **9** **18**	**3** $P_{1,0,0}$ **12**	**6** $P_{2,0,0}$ **15**
1 $P_{0,1,0}$ **10** **19**	**4** $P_{1,1,0}$ **13**	**7** $P_{2,1,0}$ **16**
2 $P_{0,2,0}$ **11** **20**	**5** $P_{1,2,0}$ **14**	**8** $P_{2,2,0}$ **17**

0 × ×
1 × ×
2 × ×
3 × ×
4 × ×
5 × ×
6 × ×
7 × ×
8 × ×
9 × ×
10 × ×
11 × ×
12 × ×
13 × ×
14 × ×
15 × ×
16 × ×
17 × ×
18 × ×
19 × ×
20 × ×
21 × ×
22 × ×
23 × ×
24 × ×
25 × ×
26 × ×

- 2-level-cyclic distribution for the rows
- cyclic distribution for the columns

0 $P_{0,0,0}$ **9** **18**	**3** $P_{1,0,0}$**12** **21**	**6** $P_{2,0,0}$**15**
1 $P_{0,1,0}$**10** **19**	**4** $P_{1,1,0}$**13**	**7** $P_{2,1,0}$**16**
2 $P_{0,2,0}$**11** **20**	**5** $P_{1,2,0}$**14**	**8** $P_{2,2,0}$**17**

0 × ×
1 × ×
2 × ×
3 × ×
4 × ×
5 × ×
6 × ×
7 × ×
8 × ×
9 × ×
10 × ×
11 × ×
12 × ×
13 × ×
14 × ×
15 × ×
16 × ×
17 × ×
18 × ×
19 × ×
20 × ×
21 × ×
22 × ×
23 × ×
24 × ×
25 × ×
26 × ×

- 2-level-cyclic distribution for the rows
- cyclic distribution for the columns

0 $P_{0,0,0}$ **9** **18**	**3** $P_{1,0,0}$**12** **21**	**6** $P_{2,0,0}$**15**
1 $P_{0,1,0}$**10** **19**	**4** $P_{1,1,0}$**13** **22**	**7** $P_{2,1,0}$**16**
2 $P_{0,2,0}$**11** **20**	**5** $P_{1,2,0}$**14**	**8** $P_{2,2,0}$**17**

0 × ×
1 × ×
2 × ×
3 × ×
4 × ×
5 × ×
6 × ×
7 × ×
8 × ×
9 × ×
10 × ×
11 × ×
12 × ×
13 × ×
14 × ×
15 × ×
16 × ×
17 × ×
18 × ×
19 × ×
20 × ×
21 × ×
22 × ×
23 × ×
24 × ×
25 × ×
26 × ×

- 2-level-cyclic distribution for the rows
- cyclic distribution for the columns

0 $P_{0,0,0}$ **9** **18**	**3** $P_{1,0,0}$**12** **21**	**6** $P_{2,0,0}$**15**
1 $P_{0,1,0}$**10** **19**	**4** $P_{1,1,0}$**13** **22**	**7** $P_{2,1,0}$**16**
2 $P_{0,2,0}$**11** **20**	**5** $P_{1,2,0}$**14** **23**	**8** $P_{2,2,0}$**17**

0 × ×
1 × ×
2 × ×
3 × ×
4 × ×
5 × ×
6 × ×
7 × ×
8 × ×
9 × ×
10 × ×
11 × ×
12 × ×
13 × ×
14 × ×
15 × ×
16 × ×
17 × ×
18 × ×
19 × ×
20 × ×
21 × ×
22 × ×
23 × ×
24 × ×
25 × ×
26 × ×

- 2-level-cyclic distribution for the rows
- cyclic distribution for the columns

0	**3**	**6**
$P_{0,0,0}$ **9**	$P_{1,0,0}$ **12**	$P_{2,0,0}$ **15**
18	**21**	**24**
1	**4**	**7**
$P_{0,1,0}$ **10**	$P_{1,1,0}$ **13**	$P_{2,1,0}$ **16**
19	**22**	
2	**5**	**8**
$P_{0,2,0}$ **11**	$P_{1,2,0}$ **14**	$P_{2,2,0}$ **17**
20	**23**	

0	×	×
1	×	×
2	×	×
3	×	×
4	×	×
5	×	×
6	×	×
7	×	×
8	×	×
9	×	×
10	×	×
11	×	×
12	×	×
13	×	×
14	×	×
15	×	×
16	×	×
17	×	×
18	×	×
19	×	×
20	×	×
21	×	×
22	×	×
23	×	×
24	×	×
25	×	×
26	×	×

- 2-level-cyclic distribution for the rows
- cyclic distribution for the columns

0	**3**	**6**
$P_{0,0,0}$ **9**	$P_{1,0,0}$ **12**	$P_{2,0,0}$ **15**
18	**21**	**24**
1	**4**	**7**
$P_{0,1,0}$ **10**	$P_{1,1,0}$ **13**	$P_{2,1,0}$ **16**
19	**22**	**25**
2	**5**	**8**
$P_{0,2,0}$ **11**	$P_{1,2,0}$ **14**	$P_{2,2,0}$ **17**
20	**23**	

0	×	×
1	×	×
2	×	×
3	×	×
4	×	×
5	×	×
6	×	×
7	×	×
8	×	×
9	×	×
10	×	×
11	×	×
12	×	×
13	×	×
14	×	×
15	×	×
16	×	×
17	×	×
18	×	×
19	×	×
20	×	×
21	×	×
22	×	×
23	×	×
24	×	×
25	×	×
26	×	×

- 2-level-cyclic distribution for the rows
- cyclic distribution for the columns

0	**3**	**6**
$P_{0,0,0}$ **9**	$P_{1,0,0}$ **12**	$P_{2,0,0}$ **15**
18	**21**	**24**
1	**4**	**7**
$P_{0,1,0}$ **10**	$P_{1,1,0}$ **13**	$P_{2,1,0}$ **16**
19	**22**	**25**
2	**5**	**8**
$P_{0,2,0}$ **11**	$P_{1,2,0}$ **14**	$P_{2,2,0}$ **17**
20	**23**	**26**

0 × ×
1 × ×
2 × ×
3 × ×
4 × ×
5 × ×
6 × ×
7 × ×
8 × ×
9 × ×
10 × ×
11 × ×
12 × ×
13 × ×
14 × ×
15 × ×
16 × ×
17 × ×
18 × ×
19 × ×
20 × ×
21 × ×
22 × ×
23 × ×
24 × ×
25 × ×
26 × ×

- 2-level-cyclic distribution for the rows
- cyclic distribution for the columns

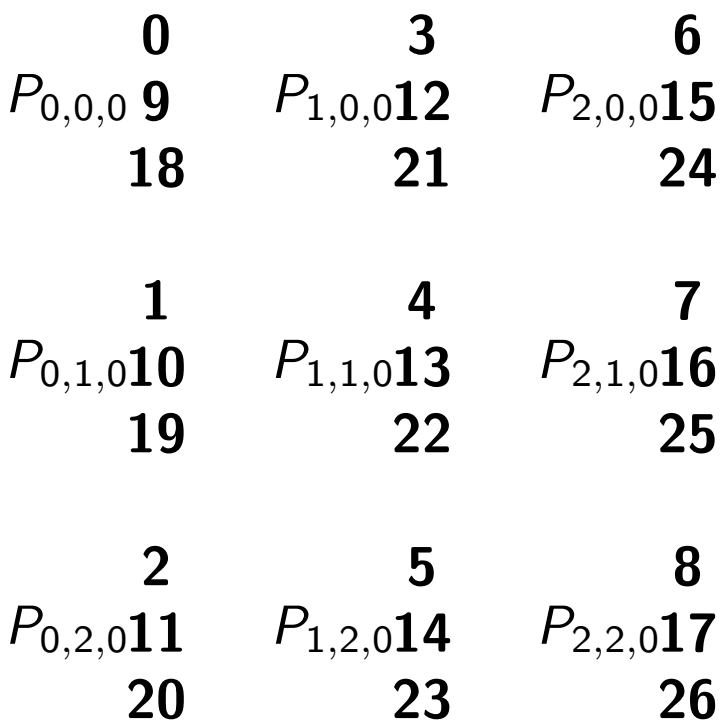
$P_{0,0,1}$ $P_{1,0,1}$ $P_{2,0,1}$

$P_{0,1,1}$ $P_{1,1,1}$ $P_{2,1,1}$

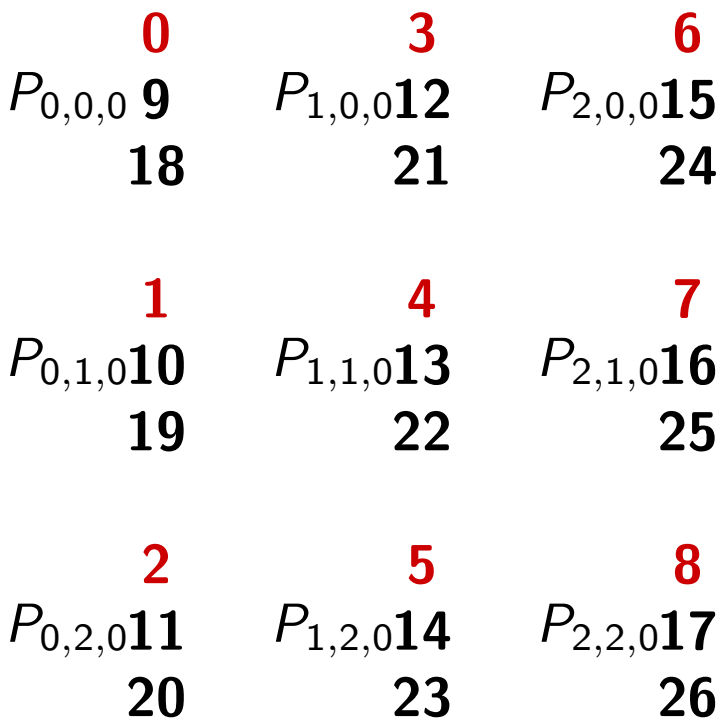
$P_{0,2,1}$ $P_{1,2,1}$ $P_{2,2,1}$

0 × ×
1 × ×
2 × ×
3 × ×
4 × ×
5 × ×
6 × ×
7 × ×
8 × ×
9 × ×
10 × ×
11 × ×
12 × ×
13 × ×
14 × ×
15 × ×
16 × ×
17 × ×
18 × ×
19 × ×
20 × ×
21 × ×
22 × ×
23 × ×
24 × ×
25 × ×
26 × ×

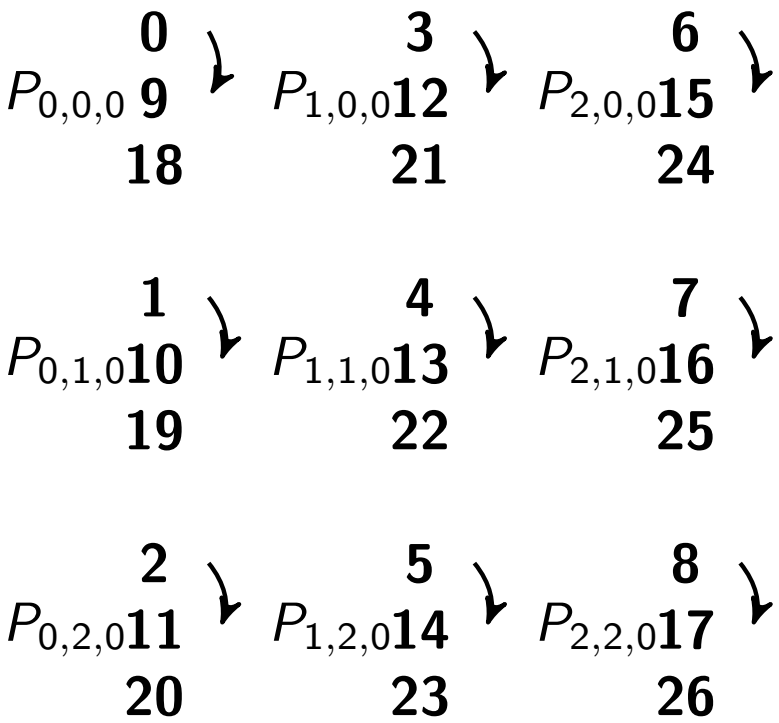
The SYSTOLIC-3D algorithm



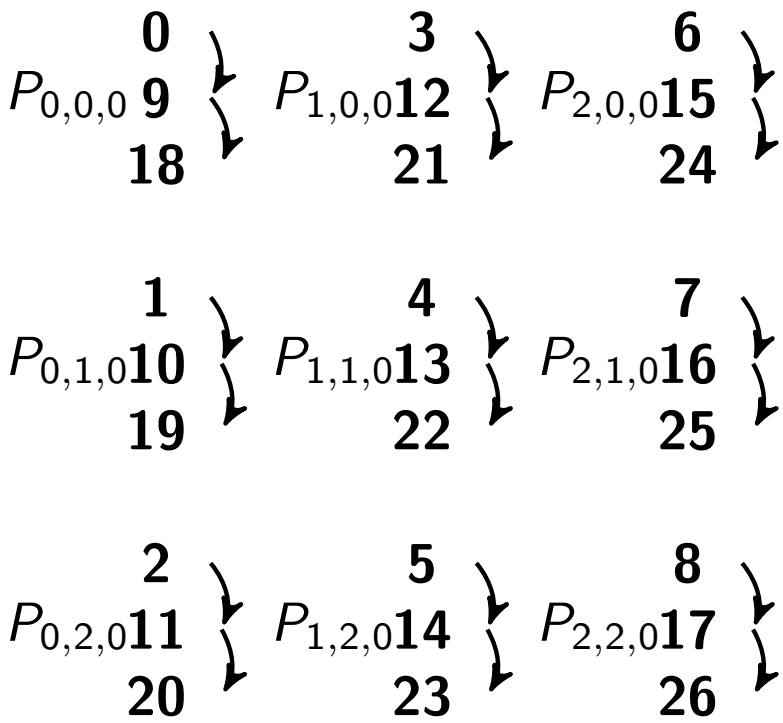
The SYSTOLIC-3D algorithm



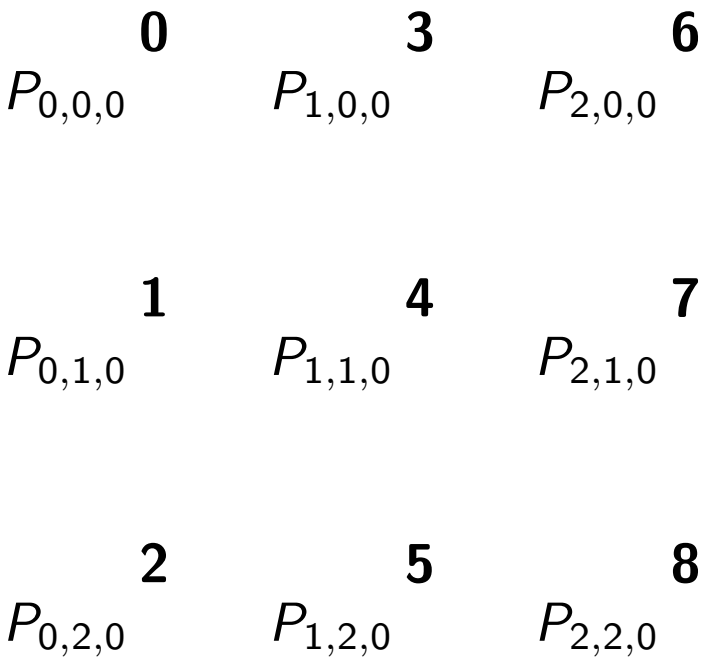
The SYSTOLIC-3D algorithm



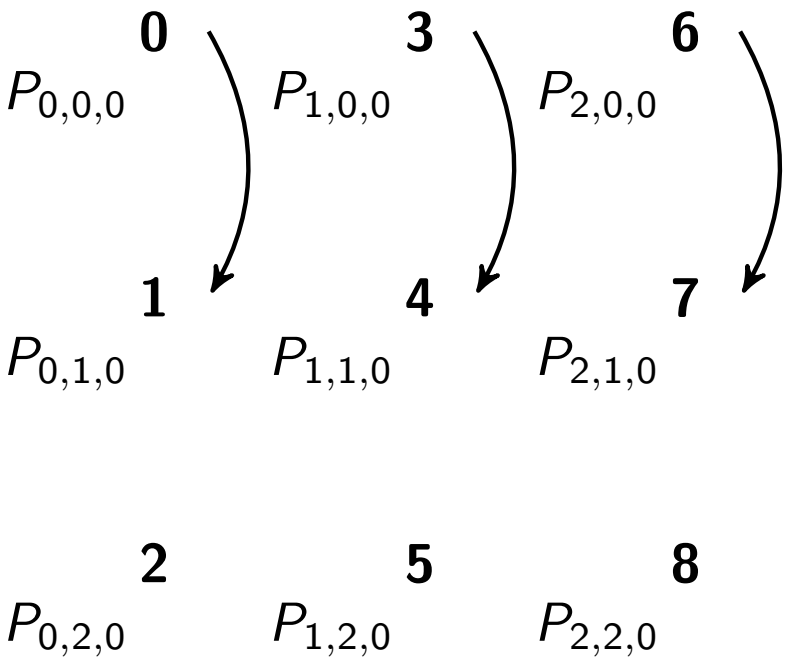
The SYSTOLIC-3D algorithm



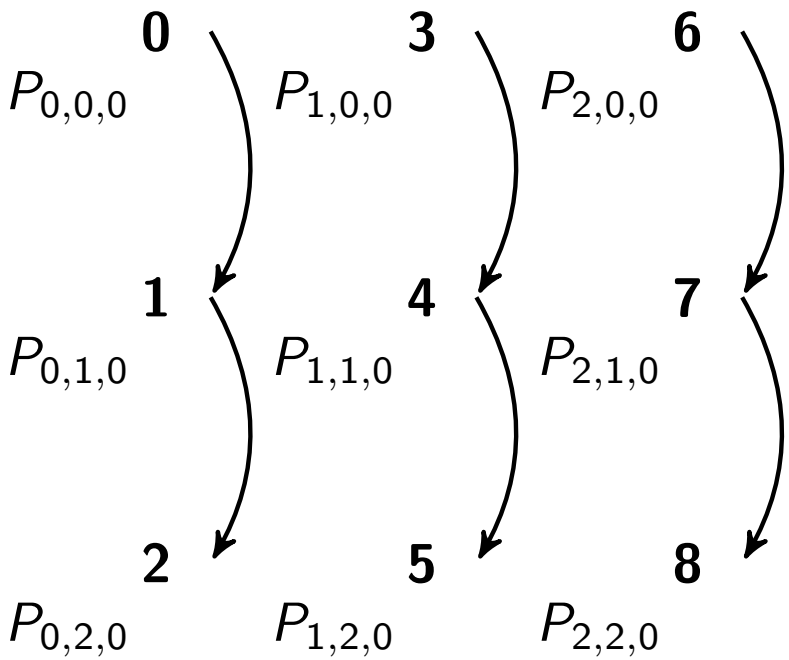
The SYSTOLIC-3D algorithm



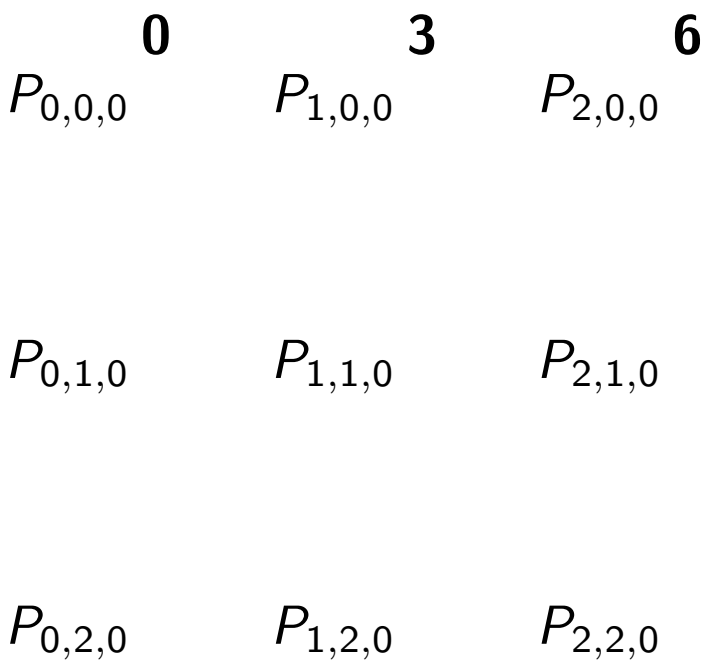
The Systolic-3D algorithm



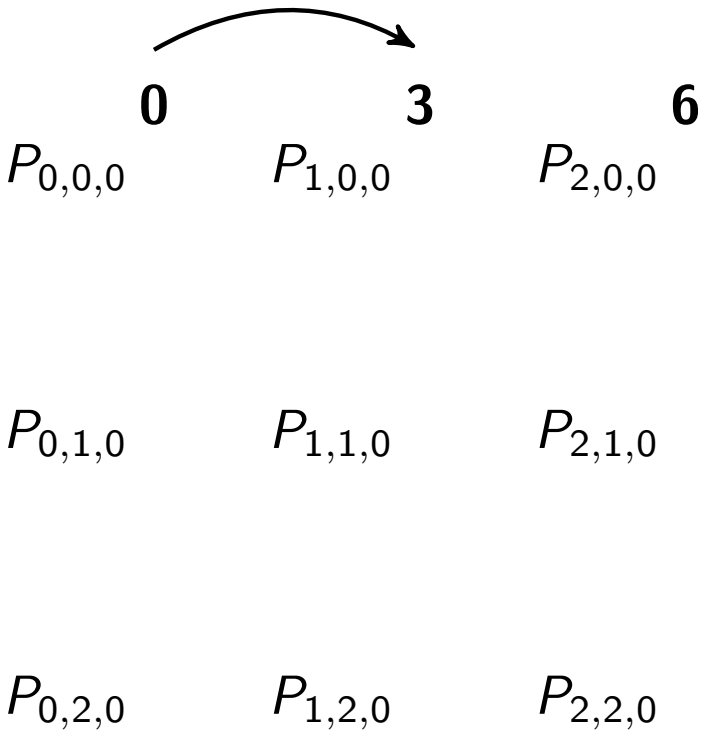
The Systolic-3D algorithm



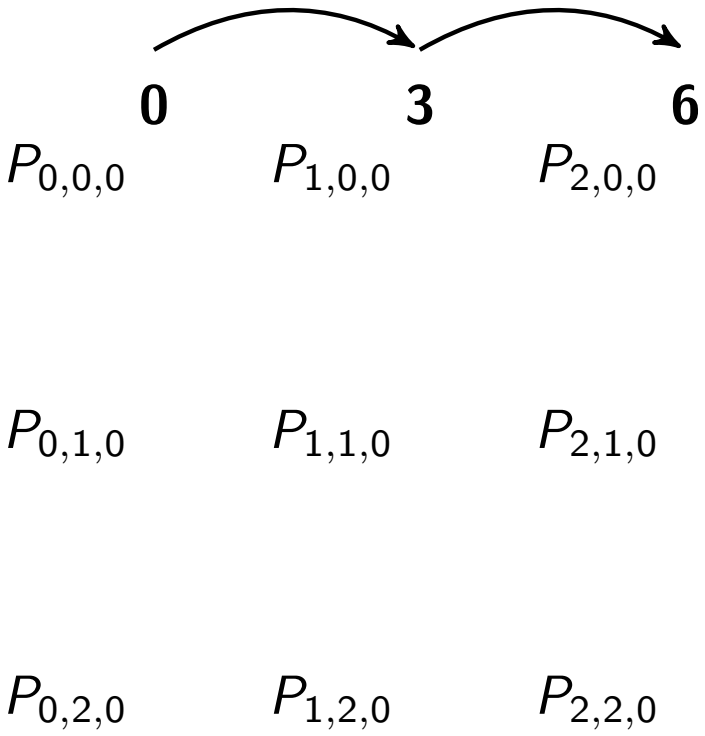
The SYSTOLIC-3D algorithm



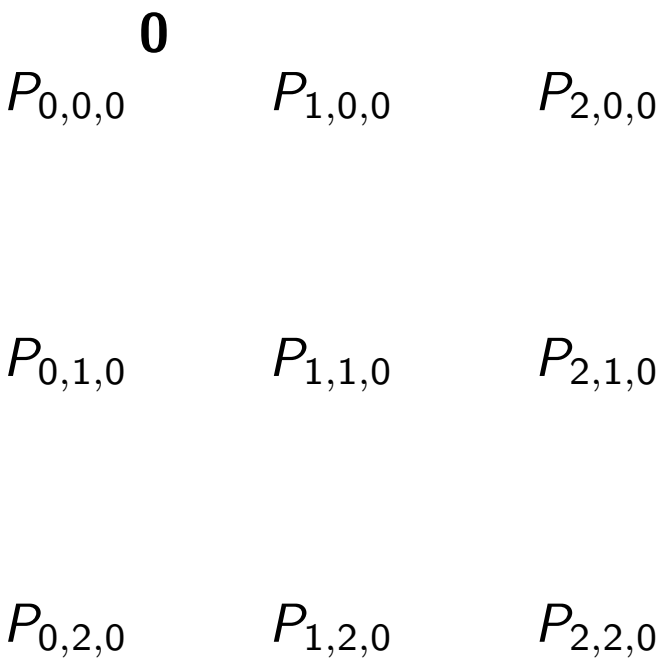
The SYSTOLIC-3D algorithm



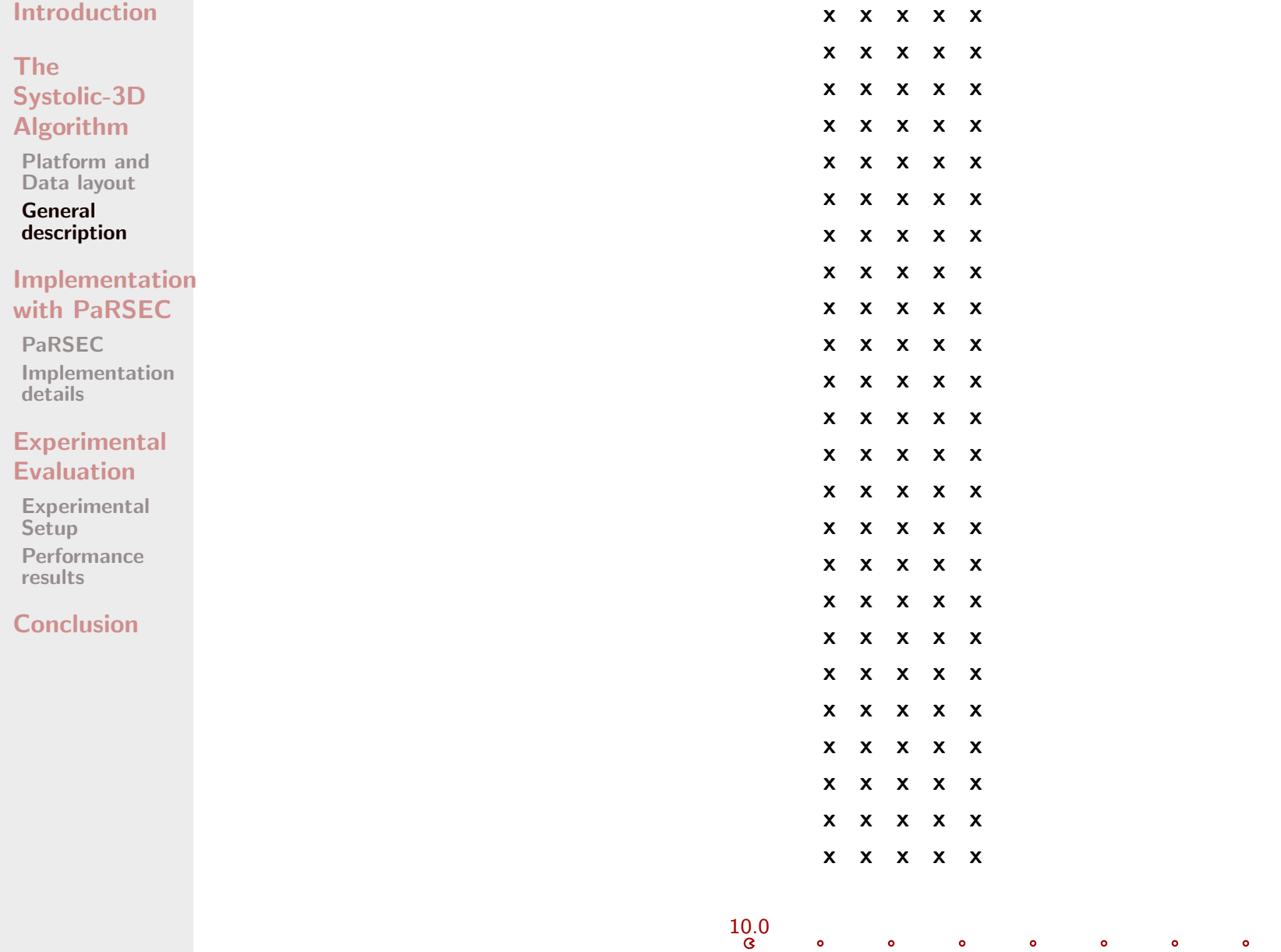
The SYSTOLIC-3D algorithm



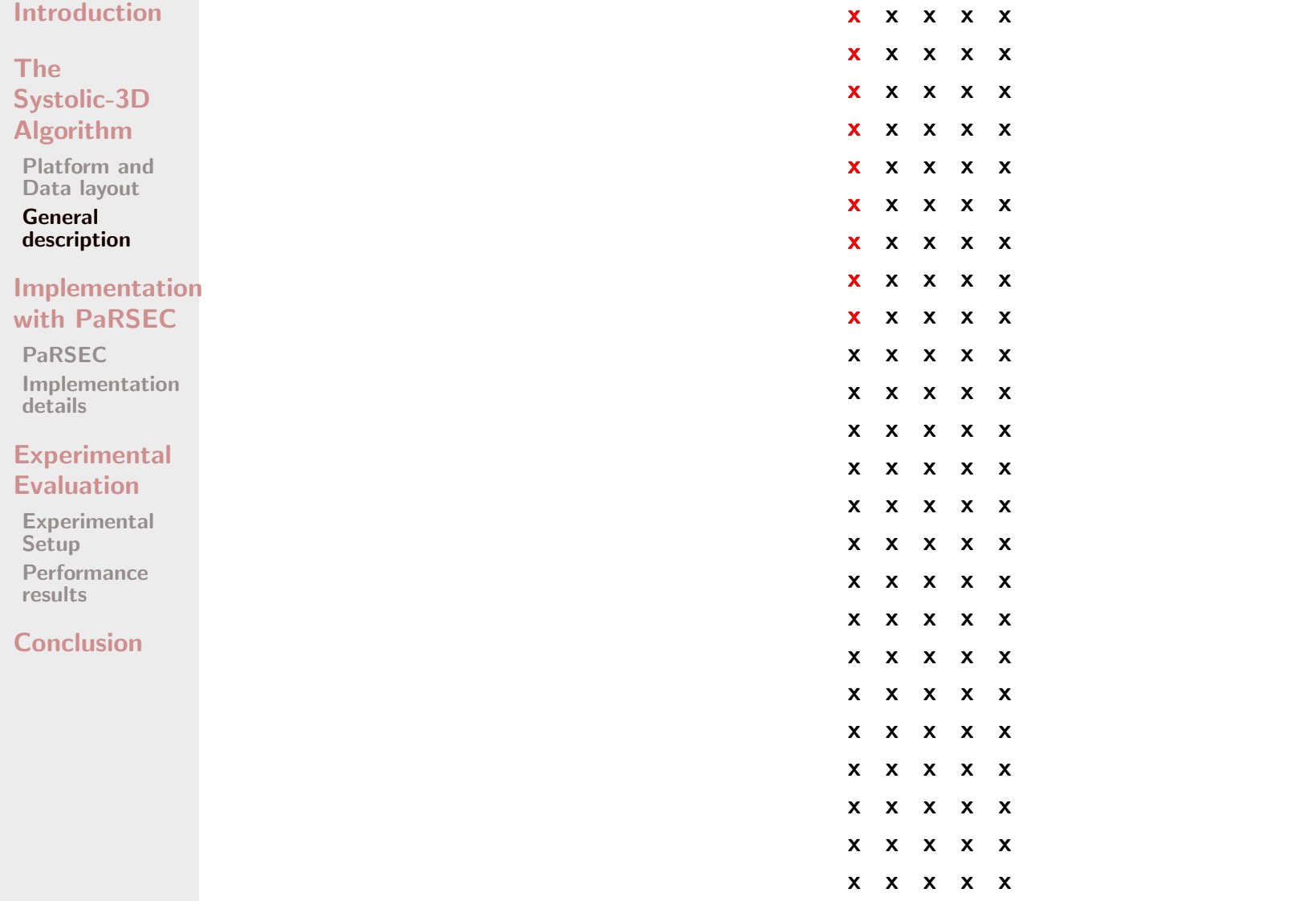
The Systolic-3D algorithm

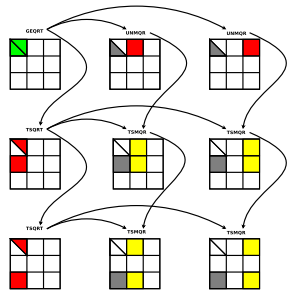


Seen differently

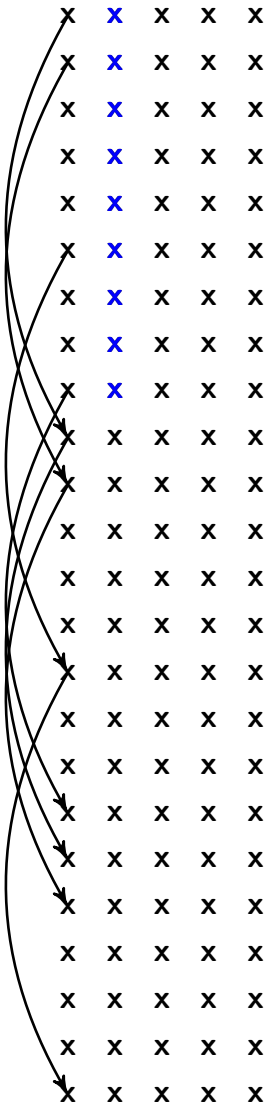


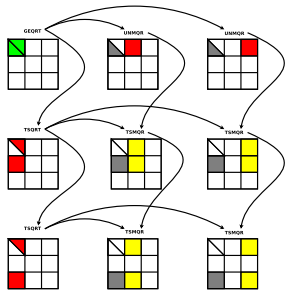
Seen differently



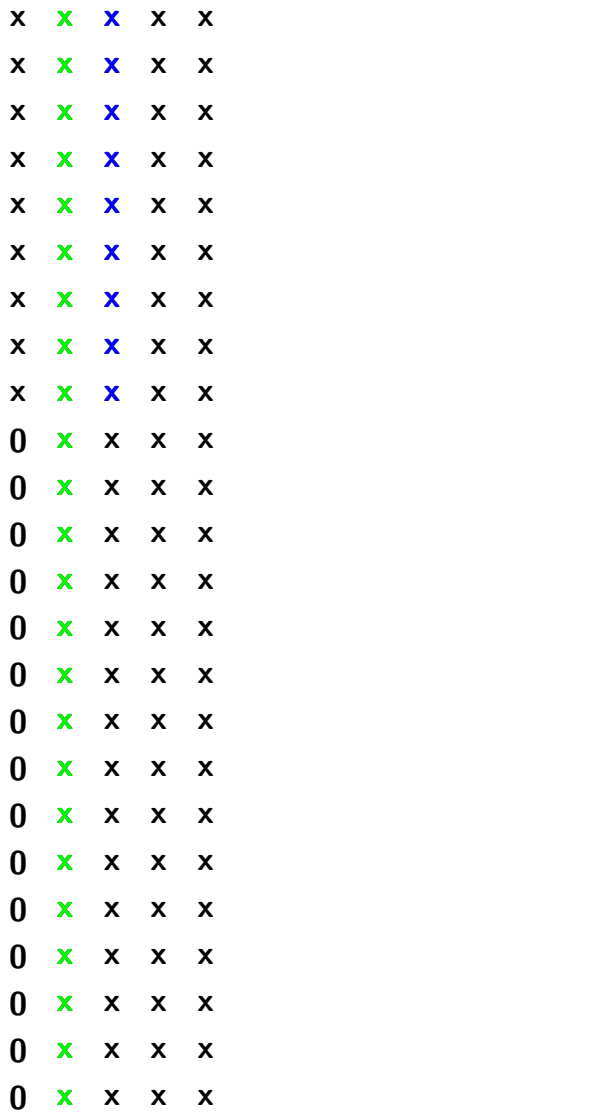


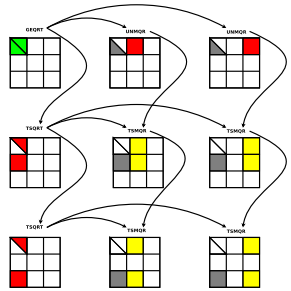
Seen differently



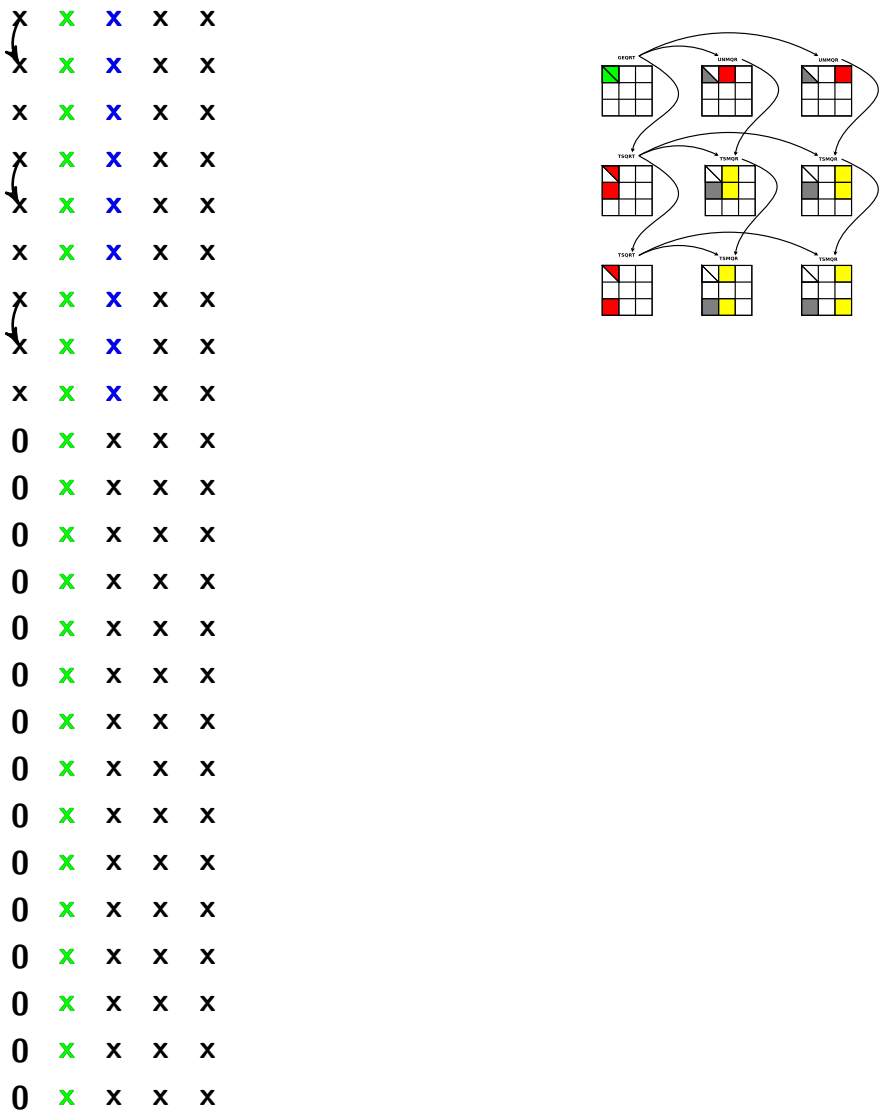


Seen differently



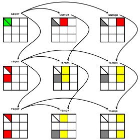


Seen differently

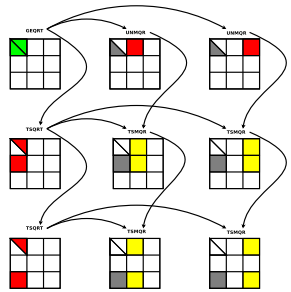


Seen differently

x	x	x	x	x
x	x	x	x	x
x	x	x	x	x
x	x	x	x	x
x	x	x	x	x
x	x	x	x	x
x	x	x	x	x
x	x	x	x	x
x	x	x	x	x
0	x	x	x	x
0	x	x	x	x
0	x	x	x	x
0	x	x	x	x
0	x	x	x	x
0	x	x	x	x
0	x	x	x	x
0	x	x	x	x
0	x	x	x	x
0	x	x	x	x
0	x	x	x	x
0	x	x	x	x
0	x	x	x	x
0	x	x	x	x
0	x	x	x	x

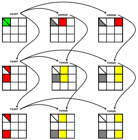


Seen differently

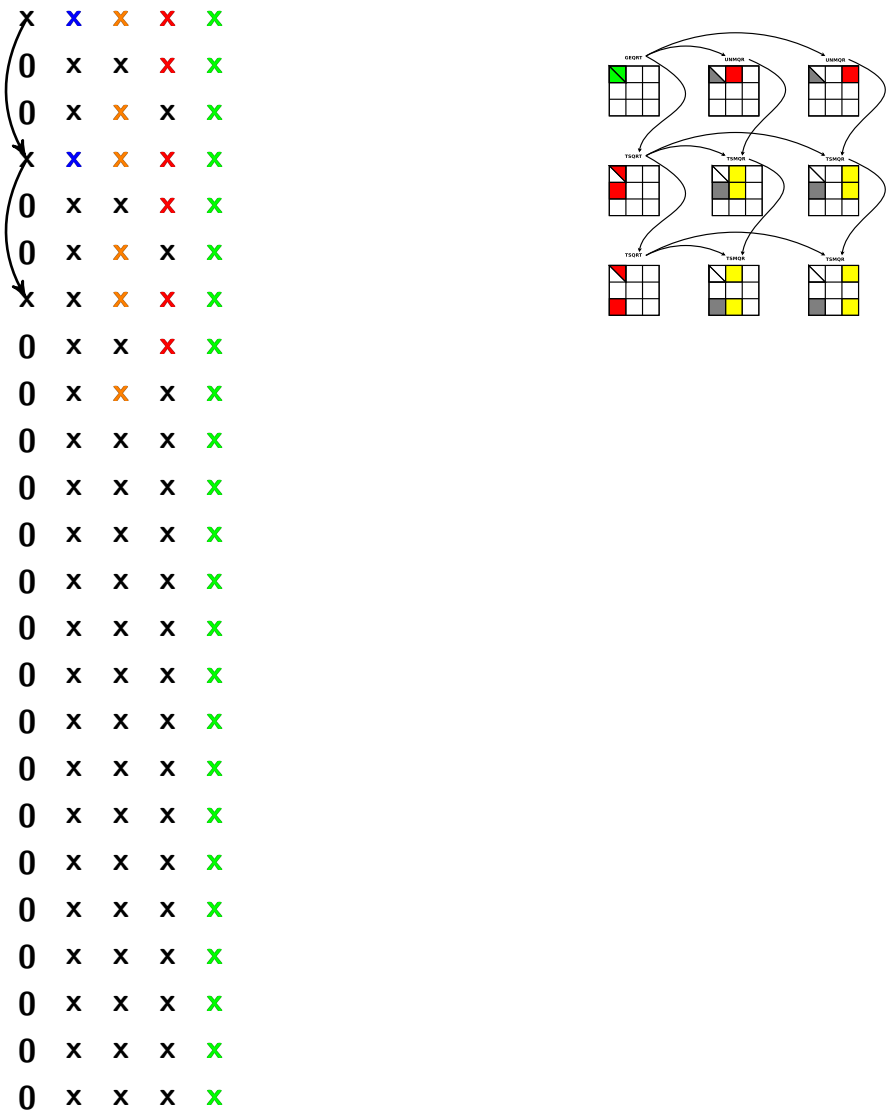


Seen differently

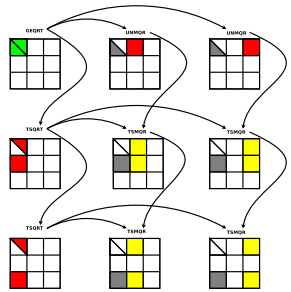
x	x	x	x	x
0	x	x	x	x
0	x	x	x	x
x	x	x	x	x
0	x	x	x	x
0	x	x	x	x
x	x	x	x	x
0	x	x	x	x
0	x	x	x	x
0	x	x	x	x
0	x	x	x	x
0	x	x	x	x
0	x	x	x	x
0	x	x	x	x
0	x	x	x	x
0	x	x	x	x
0	x	x	x	x
0	x	x	x	x
0	x	x	x	x
0	x	x	x	x
0	x	x	x	x
0	x	x	x	x
0	x	x	x	x
0	x	x	x	x



Seen differently



Seen differently



PaRSEC

- The PaRSEC environment
  ↪ we just specify the task graph
  ↪ deals with MPI communications and shared memory accesses
- It is sufficient to give an abstract representation of all the tasks (elimination and updates)
  ↪ It is sufficient to give the elimination list.

PaRSEC

- The PaRSEC environment
  ↪ we just specify the task graph
  ↪ deals with MPI communications and shared memory accesses
- It is sufficient to give an abstract representation of all the tasks (elimination and updates)
  ↪ It is sufficient to give the elimination list.

Limited effort, we only implemented a few functions that are used by PaRSEC to generate the dependency graph:

- **CurPiv**(i, k), returns the pivot for row i at step k;
- **NextPiv**(pivot, k,start), returns the next row that use "pivot" as a pivot in step k after "start";
- **PrevPiv**(pivot, k,start), returns the previous row that use "pivot" as a pivot in step k before "start";

Using these functions, PaRSEC is able to construct a dependency graph between the different tiles in order to run the algorithm as efficiently as possible.

Limited effort, we only implemented a few functions that are used by PaRSEC to generate the dependency graph:

- **CurPiv**(i, k), returns the pivot for row i at step k;
- **NextPiv**(pivot, k,start), returns the next row that use "pivot" as a pivot in step k after "start";
- **PrevPiv**(pivot, k,start), returns the previous row that use "pivot" as a pivot in step k before "start";

Using these functions, PaRSEC is able to construct a dependency graph between the different tiles in order to run the algorithm as efficiently as possible.

Kraken Supercomputer

All runs were done on the Kraken supercomputer at the National Institute for Computational Science.

- a Cray XT5 system
- 9048 computing nodes that each contain
  - two 2.6 GHz sixcore AMD Opteron (Istanbul) processors,
  - 16 GB of memory and the Cray SeaStar2+ interconnect.
- the experiments presented here used up to 1989 nodes ($\approx$ one fifth of the machine).

Matrix sizes

- small matrices, of size $M = N = 10,368$;
- medium matrices, of size $M = N = 20,736$;
- large matrices, of size $M = N = 41,472$.

Algorithms tested

- SYSTOLIC-3D
- HQR which was also implemented using the PaRSEC software, with different variants:

  HQR-FLAT uses the FLATTREE reduction;
  HQR-FIBO uses the FIBONACCI reduction;
  HQR-BINARY uses the BINARYTREE reduction;
  HQR-GREEDY uses the GREEDY reduction.

  Note that HQR uses a 2D-processor grid, we use T nodes configured as a $(pq) \times r$ 2D grid.
- SYSTOLIC-2D is a variant of SYSTOLIC-3D where q is set to 1 and then runs on a 2D grid of size $(pq) \times r$.

Algorithms tested

There is no guarantee, that the nodes assigned to the experiment will form the desired 3D torus. To the best of our knowledge, the only way to guarantee it is by booking the entire platform.

Algorithms tested

There is no guarantee, that the nodes assigned to the experiment will form the desired 3D torus. To the best of our knowledge, the only way to guarantee it is by booking the entire platform.

For each number of node T, we had to define ourselves the values of (p, q, r).

Algorithms tested

We compare with the following algorithms from the literature on the very same hardware:

- SYSTOLIC-1D a virtual systolic array decomposition. As its name indicates, it targets a 1D-linear array of processors.
- HPL $4/3$ N^3 is the virtual performance of the High Performance Linpack LU factorization using the flops count of QR: $O(\frac{4}{3}N^3)$.
- LIBSCI QR is the QR factorization from ScaLAPACK used in the Cray Scientific Library.
- HPL $2/3$ N^3 is the High Performance Linpack LU factorization with the actual flops count of LU: $O(\frac{2}{3}N^3)$.

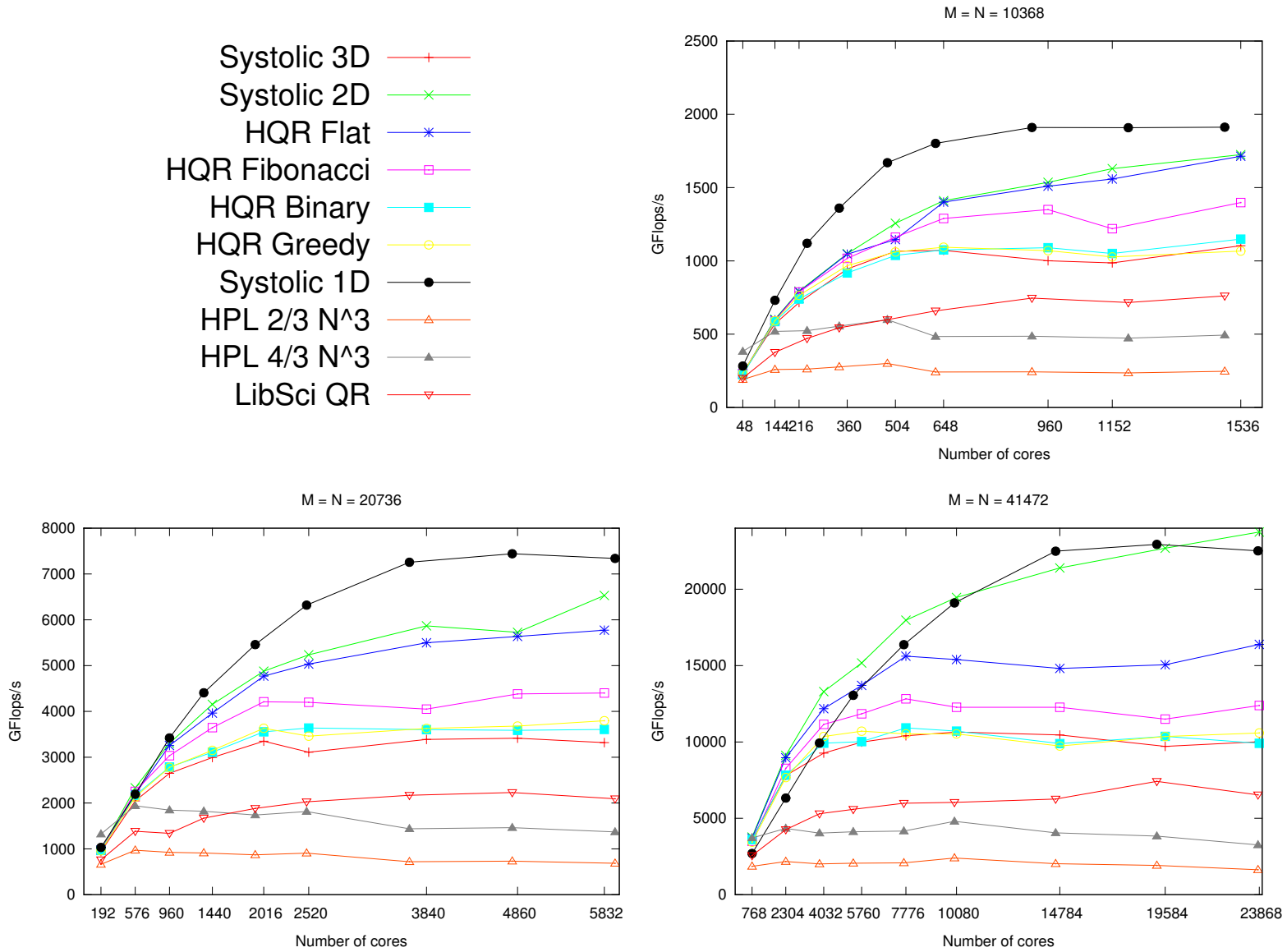


Figure : Performance of the various algorithms for different problem sizes.

Conclusion

- A systolic QR factorization algorithm, SYSTOLIC-3D, aiming at minimizing communications in reduction trees
- Main limitation to fully validate the experiments is the lack of possibility to reserve an actual 3D torus architecture on the Kraken supercomputer.
- Performance of SYSTOLIC-3D, together with its 2D counterpart, is very encouraging:
  $\rightarrow$ dramatically outperforms LIBSCI QR and HPL $4/3\ N^3$, the vendor QR factorization implementations on Kraken, and also HPL $2/3\ N^3$, the widely-used LU factorization routine (despite its favorable flop count)

Conclusion

PaRSEC is easy to use!

It enabled us to experiment with complex, hierarchical QR algorithms, without paying the price of lengthy and complex development efforts in distributed memory software engineering.

Thanks for listening. Enjoy the buffet!