Data Distribution Schemes for Dense Factorization on Any Number of Nodes

Olivier Beaumont
Jean-Alexandre Collin
Lionel Eyraud-Dubois
Mathieu Vérité

LaBRI, Inria Center of the University of Bordeaux

Scalable HPC Ecosystem & Solharis meeting
March 31st 2023

Table of Contents

Table of Contents

Part of Mathieu Vérité's PhD work

Context of the PhD

- Funded by **Région Aquitaine** – *HPC Scalable Ecosystem* project
- Data allocation for **distributed** linear algebra – followup from *Solhar* ANR project

List of publications

1. O. Beaumont, L. Eyraud-Dubois, and M. Verite. 2D Static Resource Allocation for Compressed Linear Algebra and Communication Constraints. In *IEEE HIPC 2020*, (virtual), India, Dec. 2020.
2. O. Beaumont, L. Eyraud-Dubois, J. Langou, and M. Vérité. I/O-optimal algorithms for symmetric linear algebra kernels. In *ACM SPAA 2022*, Philadephia, USA, 2022.
3. O. Beaumont, P. Duchon, L. Eyraud-Dubois, J. Langou, and M. Vérité. Symmetric Block-Cyclic Distribution: Fewer Communications Leads to Faster Dense Cholesky Factorization. In *SC 2022*, Dallas, Texas, USA, Nov. 2022.
   Best paper candidate (Algorithms track)
4. O. Beaumont, J.-A. Collin, L. Eyraud-Dubois, and M. Vérité. Data Distribution Schemes for Dense Linear Algebra Factorizations on Any Number of Nodes. In *IEEE IPDPS 2023*, St. Petersburg, Florida, USA, May 2023.

Mathieu is now a postdoc with **Laura Grigori**, in **Inria Paris** and **Sorbonne University**.

Introduction

Context

- Use cases: Dense **LU / Cholesky factorization**
- **distributed** execution using P **identical** nodes

Communications in distributed settings

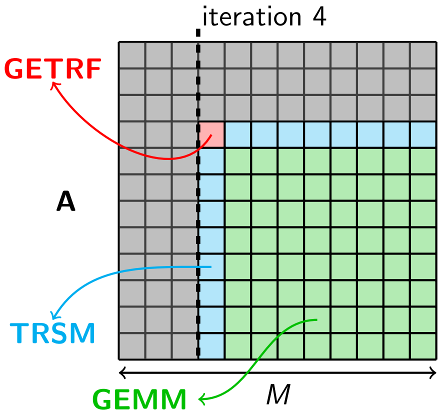
- they are a bottleneck for the execution $\Rightarrow$ reducing them improves performance
- **Approach:** design data distributions that reduce the **overall volume of communication**
- standard solution (ScaLAPACK): 2D Block-Cyclic – best when P is square
- for symmetric input: Symmetric Block Cyclic (SBC) – valid for limited values of P

In this talk

- Design distributions for **any** number of nodes

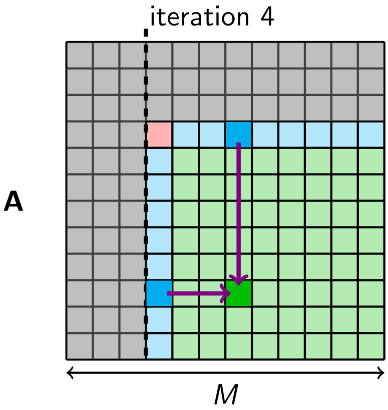
Table of Contents

Communication Scheme in Distributed LU factorization



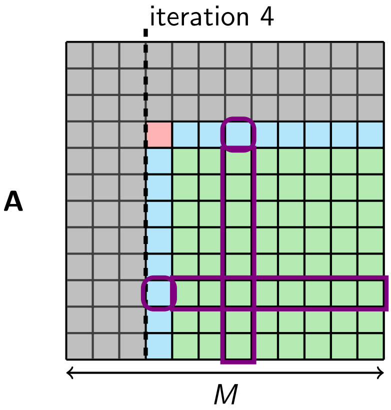
- Dominant part of the communication: TRSM output $\rightarrow$ GEMM input.

Communication Scheme in Distributed LU factorization



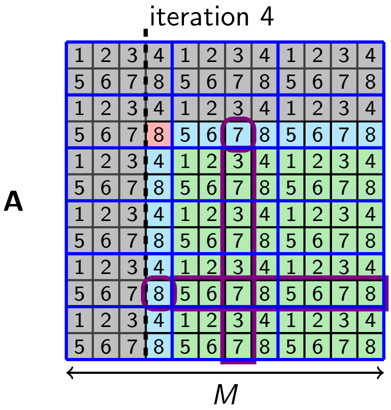
- Dominant part of the communication: TRSM output → GEMM input.

Communication Scheme in Distributed LU factorization



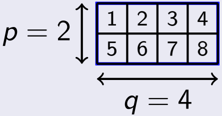
- Dominant part of the communication: TRSM output $\rightarrow$ GEMM input.

Communication Scheme in Distributed LU factorization



- Dominant part of the communication: TRSM output $\rightarrow$ GEMM input.

With the 2D Block Cyclic Pattern

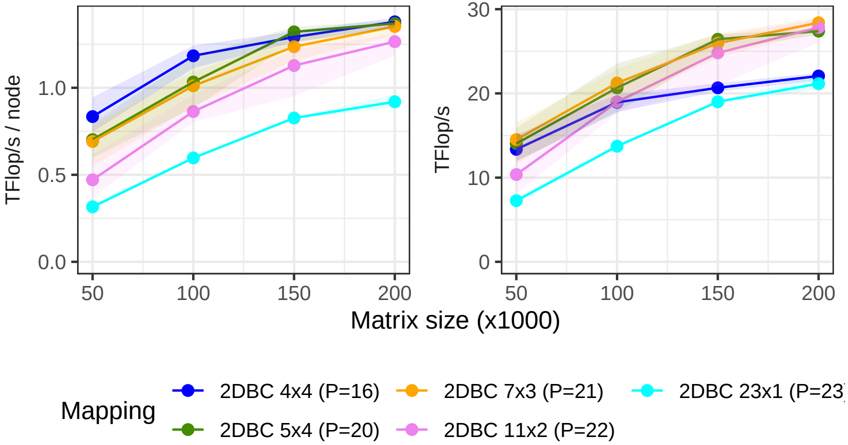


- Each tile of the lower triangular is sent $q-1$ times
- Each tile of the upper triangular is sent $p-1$ times
- Total communication cost: $Q = \frac{M(M+1)}{2}(p+q-2)$
- Best when $p \simeq q \simeq \sqrt{P}$

What if $P = 23$? (LU nopiv with Chameleon+StarPU)

Common answer: just use fewer nodes to get a nice 2DBC distribution

Experiments on `bora` nodes of Plafrim: 36-core Intel Xeon Skylake Gold 6240 @ 2.6 GHz



What if $P = 23$? [IPDPS'23]

Can we try to arrange the nodes in a square?

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

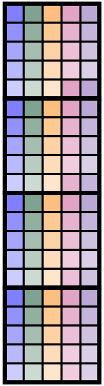
Can we try to arrange the nodes in a square?

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

What if $P = 23$? [IPDPS'23]

Can we try to arrange the nodes in a square?

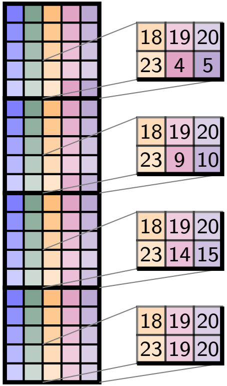
1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23		



What if $P = 23$? [IPDPS'23]

Can we try to arrange the nodes in a square?

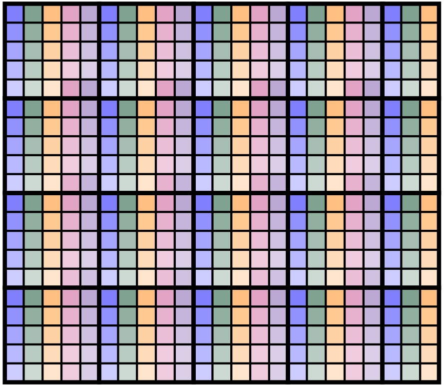
1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23		



Can we try to arrange the nodes in a square?

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

G-2DBC: a 20×23 pattern
Each node appears 20 times.
5 nodes in each row,
4 or 5 in each column.



With **G-2DBC**, one can use **all** nodes with **good efficiency**

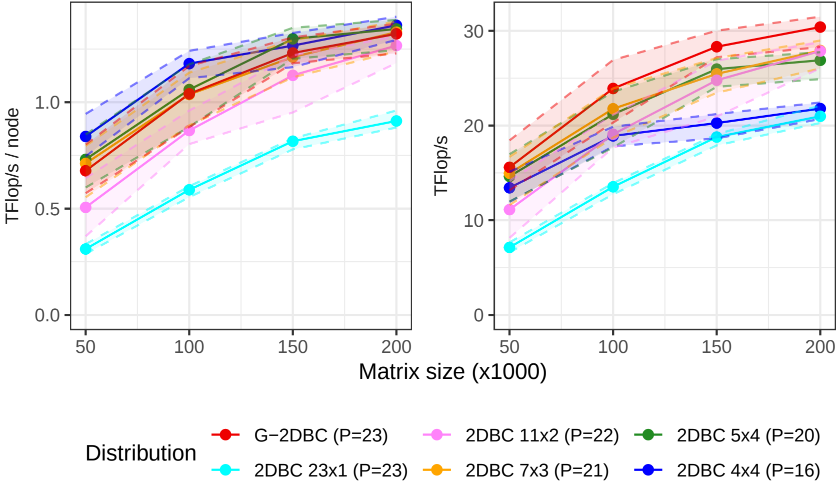
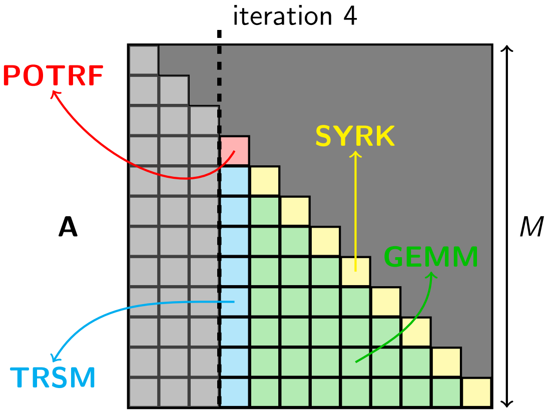


Table of Contents

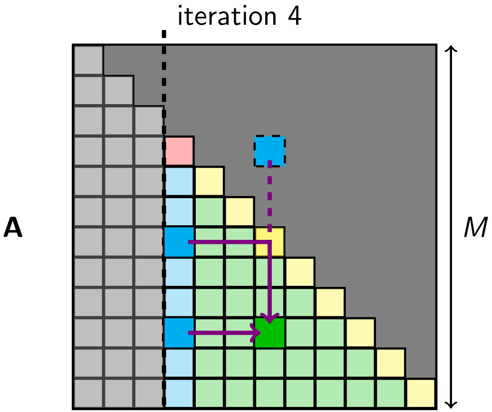
Communication Scheme in Distributed Cholesky



- Dominant part of the communication: TRSM output → GEMM input.
- Symmetry of **A** ⇒ as many transfers as **different nodes** in the **union** of a row and column.

- The union of row and column of same index: **ColRow**.
- Criterion for communication reduction: **number of different nodes** in ColRow: for $i \in \{1, \ldots, M\}$, it is denoted z_i.

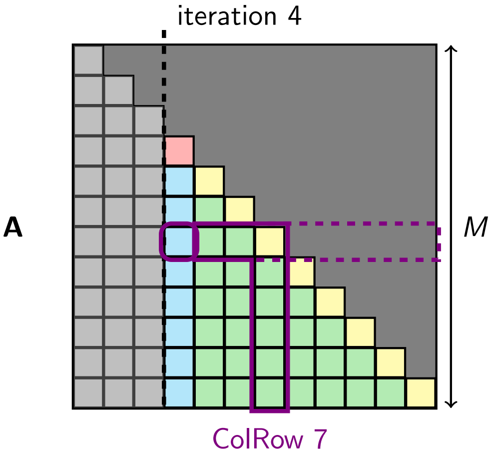
Communication Scheme in Distributed Cholesky



- Dominant part of the communication: TRSM output → GEMM input.
- Symmetry of **A** ⇒ as many transfers as **different nodes** in the **union** of a row and column.

- The union of row and column of same index: **ColRow**.
- Criterion for communication reduction: **number of different nodes** in ColRow: for $i \in \{1, \ldots, M\}$, it is denoted z_i.

Communication Scheme in Distributed Cholesky



- Dominant part of the communication: TRSM output → GEMM input.
- Symmetry of **A** ⇒ as many transfers as **different nodes** in the **union** of a row and column.

- The union of row and column of same index: **ColRow**.
- Criterion for communication reduction: **number of different nodes** in ColRow: for $i \in \{1, \ldots, M\}$, it is denoted z_i.

Communication Cost of Pattern-based Distributions

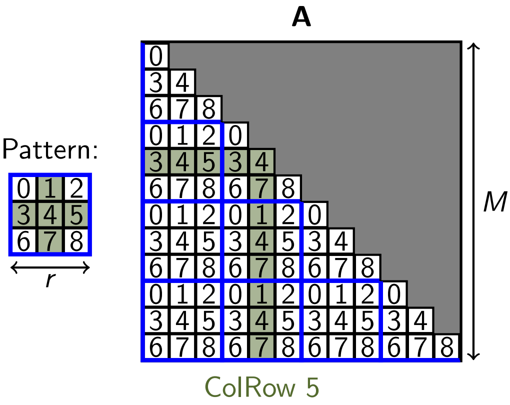


Figure: 2D BC distribution using $P = 9$ nodes.

Square pattern $\Rightarrow$ **matching ColRow** in the matrix and the pattern.

At iteration k:

- pattern replicated vertically $\frac{M-k}{r}$ times
- each node in column k broadcasts to all other nodes in its ColRow

$\Rightarrow$ #comm $= (M-k)\left(\frac{1}{r}\sum_{i=1}^{r} z_i - 1\right)$

Total volume of communication:

$$Q = \underbrace{\frac{M(M+1)}{2}}_{\text{size of } \mathbf{A}} \Big(\underbrace{\frac{1}{r}\sum_{i=1}^{r} z_i}_{\text{pattern comm cost: } \bar{z}} - 1 \Big)$$

Communication Cost of Pattern-based Distributions

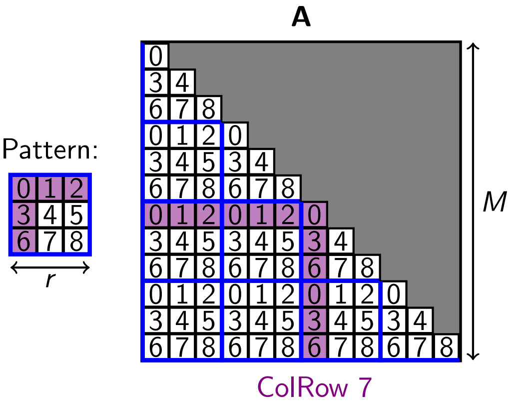


Figure: 2D BC distribution using $P = 9$ nodes.

Square pattern $\Rightarrow$ **matching ColRow** in the matrix and the pattern.

At iteration k:

- pattern replicated vertically $\frac{M-k}{r}$ times
- each node in column k broadcasts to all other nodes in its ColRow

$\Rightarrow$ #comm $= (M-k)\left(\frac{1}{r}\sum_{i=1}^{r} z_i - 1\right)$

Total volume of communication:

$$Q = \underbrace{\frac{M(M+1)}{2}}_{\text{size of } \mathbf{A}} \Big(\underbrace{\frac{1}{r}\sum_{i=1}^{r} z_i}_{\text{pattern comm cost: } \bar{z}} - 1 \Big)$$

Communication Cost of Pattern-based Distributions

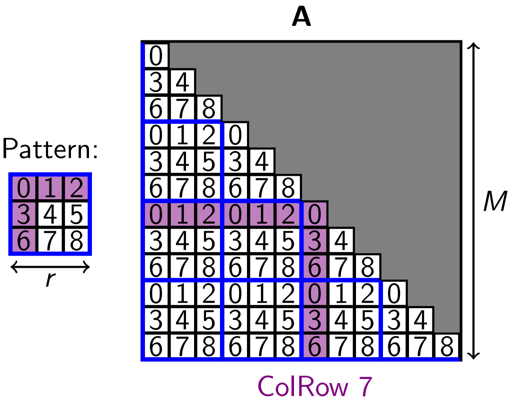


Figure: 2D BC distribution using $P = 9$ nodes.

Q only depends on the **pattern communication cost** (*i.e.* "average number of different nodes per ColRow ")

$$\bar{z} = \frac{1}{r} \sum_{i=1}^{r} z_i$$

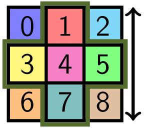
Objective: minimize it.

Symmetric patterns are good candidates: same nodes on rows and columns.

Constraint: pattern must be **balanced** (each node appears the same number of times)

Communication Cost: BC, SBC and TBC

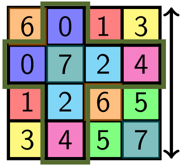
2D BC pattern ($P = 9$):



$r = 3 \; (= \sqrt{P})$

#nodes in ColRow 2:
$5 \; (= 2\sqrt{P} - 1)$

SBC *basic* pattern ($P = 8$):



$r = 4 \; (= \sqrt{2}\sqrt{P})$

#nodes in ColRow 2:
$4 \; (= \sqrt{2}\sqrt{P})$

2D Block Cyclic (BC)

- balanced: each node appears once
- size $r = \sqrt{P}$ (smallest possible with P)
- communication cost: $\bar{z} = 2r - 1 = 2\sqrt{P} - 1$

Symmetric Block Cyclic (SBC)

Triangular Block Cyclic (TBC)

Communication Cost: BC, SBC and TBC

2D BC pattern ($P = 9$):

0	1	2
3	4	5
6	7	8

$r = 3\ (= \sqrt{P})$

#nodes in ColRow 2:
$5\ (= 2\sqrt{P} - 1)$

SBC *basic* pattern ($P = 8$):

6	0	1	3
0	7	2	4
1	2	6	5
3	4	5	7

$r = 4\ (= \sqrt{2}\sqrt{P})$

#nodes in ColRow 2:
$4\ (= \sqrt{2}\sqrt{P})$

2D Block Cyclic (BC)

$$\bar{z} = 2\sqrt{P} - 1$$

Symmetric Block Cyclic (SBC)

- $\frac{r(r-1)}{2}$ nodes below diagonal
- $\frac{r}{2}$ nodes on the diagonal $\Rightarrow P = \frac{r^2}{2}$
- balanced: each node appears 2 times
- smallest symmetric version (larger than BC)
- communication cost: $\bar{z} = r = \sqrt{2}\sqrt{P}$

Triangular Block Cyclic (TBC)

Communication Cost: BC, SBC and TBC

TBC pattern ($P = 12$)

	1	1	4	5	6	4	5	6
1		1	7	8	9	9	7	8
1	1		10	11	12	11	12	10
4	7	10		2	2	4	7	10
5	8	11	2		2	11	5	8
6	9	12	2	2		9	12	6
4	9	11	4	11	9		3	3
5	7	12	7	5	12	3		3
6	8	10	10	8	6	3	3	

$r = 9 (\simeq P)$

#nodes in ColRow 2:
$4 \ (\simeq \sqrt{P} + 0.5)$

2D Block Cyclic (BC)

$$\bar{z} = 2\sqrt{P} - 1$$

Symmetric Block Cyclic (SBC)

$$\bar{z} = \sqrt{2}\sqrt{P}$$

Triangular Block Cyclic (TBC)

- larger and more complex pattern
- $r = c^2$ for any **prime** number c
- $P = c(c+1)$
- $\bar{z} = c + 1 = \frac{1}{2} + \sqrt{P + \frac{1}{4}}$

Communication Cost: BC, SBC and TBC

TBC pattern ($P = 12$)

	1	1	4	5	6	4	5	6
1		1	7	8	9	9	7	8
1	1		10	11	12	11	12	10
4	7	10		2	2	4	7	10
5	8	11	2		2	11	5	8
6	9	12	2	2		9	12	6
4	9	11	4	11	9		3	3
5	7	12	7	5	12	3		3
6	8	10	10	8	6	3	3	

$r = 9 (\simeq P)$

#nodes in ColRow 2:
$4 \ (\simeq \sqrt{P} + 0.5)$

2D Block Cyclic (BC)

$$\bar{z} = 2\sqrt{P} - 1$$

Symmetric Block Cyclic (SBC)

$$\bar{z} = \sqrt{2}\sqrt{P}$$

Triangular Block Cyclic (TBC)

$$\bar{z} = \frac{1}{2} + \sqrt{P + \frac{1}{4}}$$

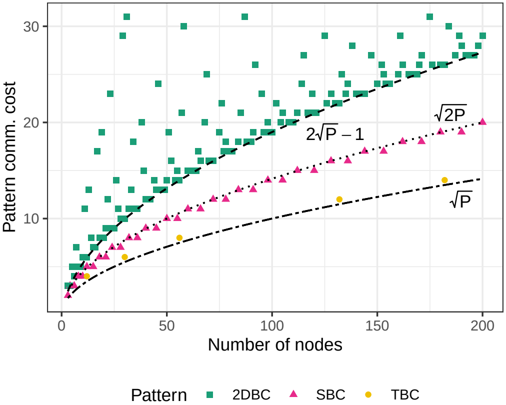
Asymptotically, SBC **reduces comms by a factor of** $\sqrt{2}$. TBC reduces by **another** $\sqrt{2}$ factor.

SBC and TBC Limitations

Not available for any P

r/c	SBC basic	SBC extended	TBC
3	-	3	12
4	8	6	-
5	-	10	30
6	18	15	-
7	-	21	56
8	32	28	-
9	-	36	-
10	50	45	-

$\Rightarrow$ What to do with $P = 35$?



Greedy ColRow & Matching (GCR&M)

General ideas

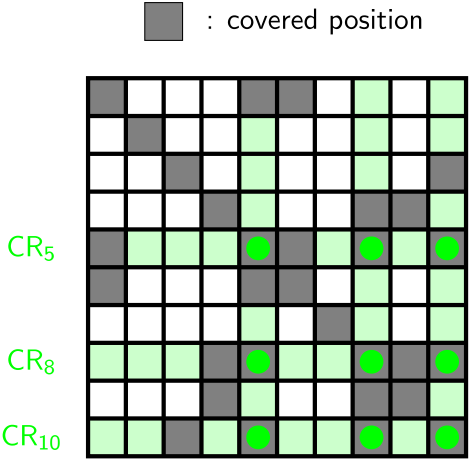
- look for **larger** symmetric pattern
- minimize $\bar{z}$ under constraint of almost perfect balancing (excluding diagonal)
- **diagonal** positions unallocated $\rightarrow$ used to **compensate imbalance**

GCR&M algorithm

Input: pattern size r, number of nodes P
Output: symmetric square pattern
Two steps:

1. associate each position $\leftrightarrow$ subset of possible nodes (*greedy* procedure)
2. allocate each pattern position to a node (*matching*)

Greedy ColRow & Matching (GCR&M)



GCR&M algorithm - **step 1**

Throughout the execution, maintain:

- set of uncovered pattern positions: $\mathcal{U}$ (init. all positions, $\mathcal{U} = \{1, \dots, r\}^2$)
- for each node p, the set of ColRow in which p can appear: $\mathcal{A}[p]$

While $\mathcal{U} \neq \emptyset$:

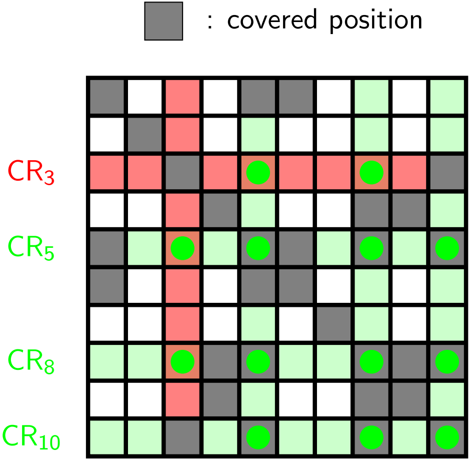
(a) select the least loaded node p

(b) assign to p the ColRow which **maximize newly covered positions**

(c) update $\mathcal{U}$

"Reverse" $\mathcal{A}$: each position $\leftrightarrow$ subset of nodes

Greedy ColRow & Matching (GCR&M)



CR $\{1, 3, 4, 6, 9\}$ cover 4 new positions

GCR&M algorithm - **step 1**

Throughout the execution, maintain:

- set of uncovered pattern positions: $\mathcal{U}$ (init. all positions, $\mathcal{U} = \{1, \ldots, r\}^2$)
- for each node p, the set of ColRow in which p can appear: $\mathcal{A}[p]$

While $\mathcal{U} \neq \emptyset$:

(a) select the least loaded node p

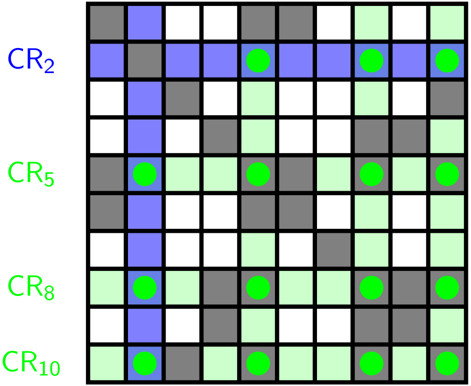
(b) assign to p the ColRow which **maximize newly covered positions**

(c) update $\mathcal{U}$

"Reverse" $\mathcal{A}$: each position $\leftrightarrow$ subset of nodes

Greedy ColRow & Matching (GCR&M)

: covered position



CR $\{2, 7\}$ cover 6 new positions

GCR&M algorithm - **step 1**

Throughout the execution, maintain:

- set of uncovered pattern positions: $\mathcal{U}$ (init. all positions, $\mathcal{U} = \{1, \ldots, r\}^2$)
- for each node p, the set of ColRow in which p can appear: $\mathcal{A}[p]$

While $\mathcal{U} \neq \emptyset$:

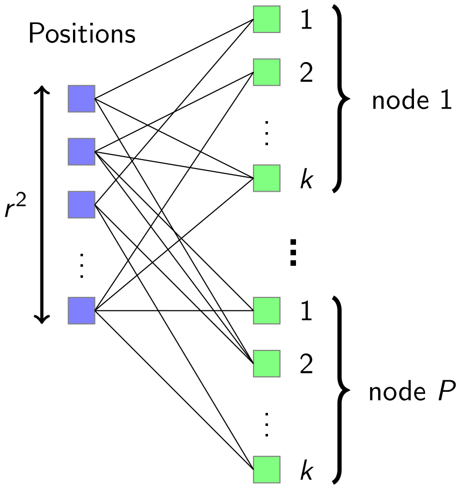
(a) select the least loaded node p

(b) assign to p the ColRow which **maximize newly covered positions**

(c) update $\mathcal{U}$

"Reverse" $\mathcal{A}$: each position $\leftrightarrow$ subset of nodes

Greedy ColRow & Matching (GCR&M)

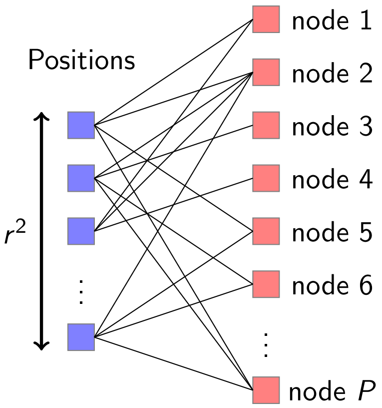


GCR&M algorithm - step 2

Association position $\leftrightarrow$ possible nodes: **bipartite graph**

- Build an allocation by finding a maximum cardinality matching in two successive versions of the graph:
  (a) using $k = \lfloor \frac{r(r-1)}{P} \rfloor$ replications of each node $\rightarrow$ ensure balancing
  (b) using 1 replication for each node
- Remaining unallocated positions $\rightarrow$ assign to the least loaded possible node

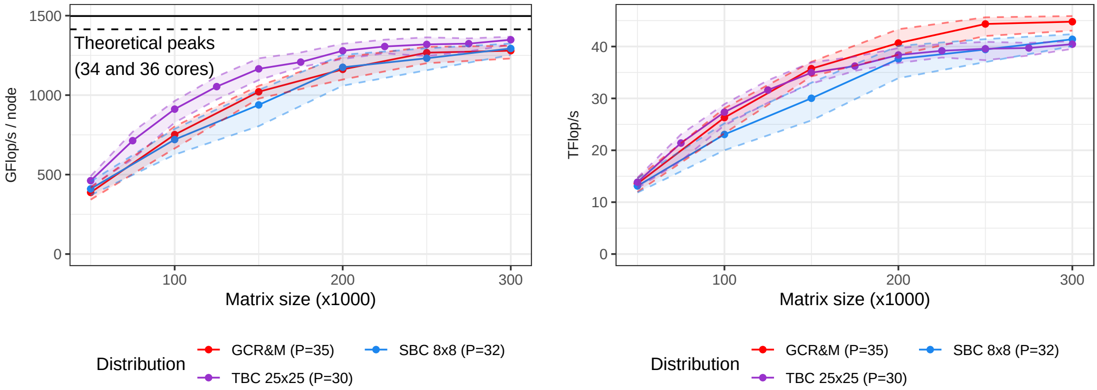
Greedy ColRow & Matching (GCR&M)



GCR&M algorithm - step 2

Association position $\leftrightarrow$ possible nodes:
bipartite graph

- Build an allocation by finding a maximum cardinality matching in two successive versions of the graph:
  (a) using $k = \lfloor \frac{r(r-1)}{P} \rfloor$ replications of each node $\rightarrow$ ensure balancing
  (b) using 1 replication for each node
- Remaining unallocated positions $\rightarrow$ assign to the least loaded possible node



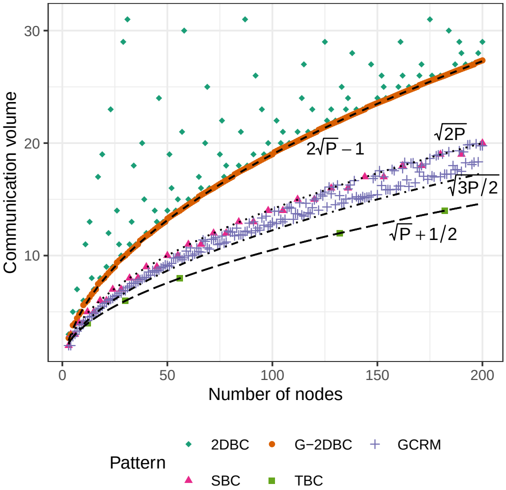
SBC (basic)	$P = 32$	$r = 8$	$\bar{z} = 8$
GCR&M	$P = 35$	$r = 15$	$\bar{z} = 7.4$
TBC	$P = 30$	$r = 25$	$\bar{z} = 6$

Table of Contents

Conclusion and Perspectives

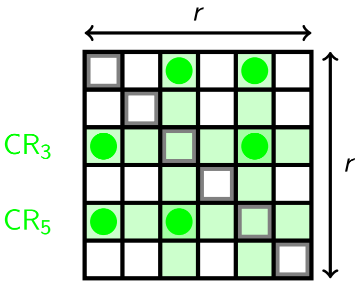
Achievements

- Generic G-2DBC pattern
- GCR&M easy and fast
- can provide patterns for any P "offline"
- matches or improves over 2DBC/SBC in most cases
- efficient use of any number of resources



Conclusion and Perspectives

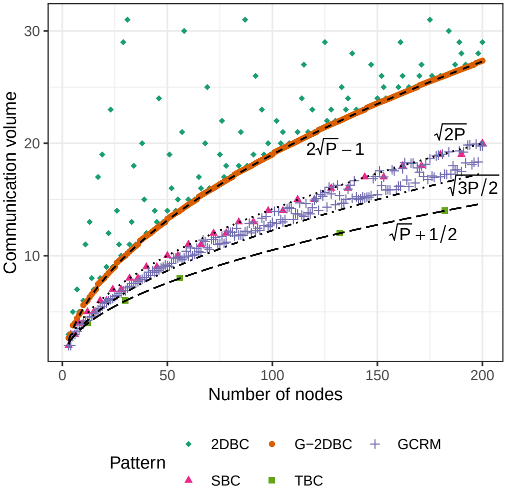
Where does $\sqrt{\frac{3}{2}}\sqrt{P}$ comes from?



In such a configuration:

$\#\text{positions} = 6P \;\Rightarrow\; r \approx \sqrt{6P}$

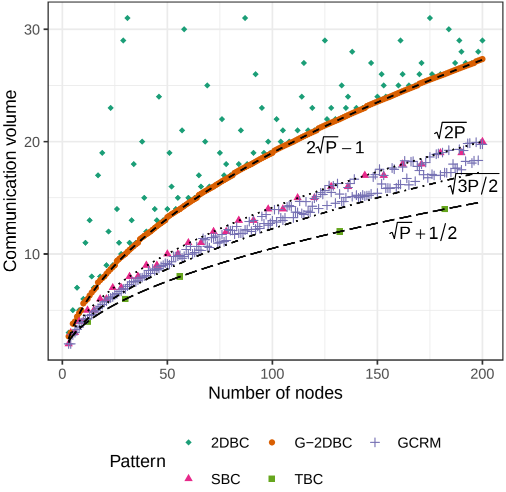
thus: $\bar{z} = \frac{r}{2} \approx \sqrt{\frac{3}{2}}\sqrt{P}$



Conclusion and Perspectives

GCR&M solution for $P = 35$:

	1	2	2	0	5	15	22	15	27	30	22	27	30	5
1		31	33	19	1	6	19	33	11	6	11	31	16	16
2	31		2	4	8	32	17	8	24	17	10	31	24	32
2	33	2		14	1	18	12	33	18	10	10	12	3	14
0	19	4	14		21	21	19	34	9	9	34	4	0	14
5	1	8	1	21		21	20	8	20	25	25	13	13	5
15	6	32	18	21	21		28	15	18	6	26	26	28	32
22	19	17	12	19	20	28		7	20	17	22	12	28	7
15	33	8	33	34	8	15	7		29	23	34	23	0	29
27	11	24	18	9	20	18	20	29		9	11	27	24	29
30	6	17	10	9	25	6	17	23	9		25	23	30	7
22	11	10	10	34	25	26	22	34	11	25		26	3	3
27	31	31	12	4	13	26	12	23	27	23	26		13	4
30	16	24	3	0	13	28	28	0	24	30	3	13		16
5	16	32	14	14	5	32	7	29	29	7	3	4	16	



$r = 15 \approx \sqrt{6P} (\approx 14.491)$
and $\bar{z} = 7.4 \approx \frac{r}{2} (= 7.5)$

Conclusion and Perspectives

Difficulties

- GCR&M algorithm is **complicated**
- better theoretical foundation: **how to choose r**
- study the effect of local imbalance

Future work

- provide a **“database“** of communication-efficient patterns for any P
- connect the underlying combinatorial problem with existing references

Thank you for your attention

Questions?