Sparstition: A Partitioning Scheme for Large-Scale Sparse Matrix Vector Multiplication on FPGA

Delft University of Technology

Björn Sigurbergsson, Tom Hogervorst, Tong Dong Qiu, Razvan Nane

15th July, 2019

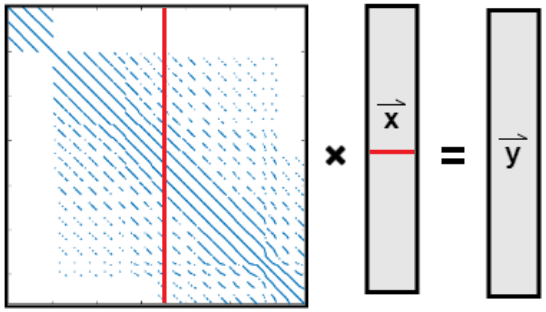
SpMV

- Sparse Matrix Vector Multiplication (SpMV or SMVM).
  - Important kernel found in many **iterative** applications.

SpMV

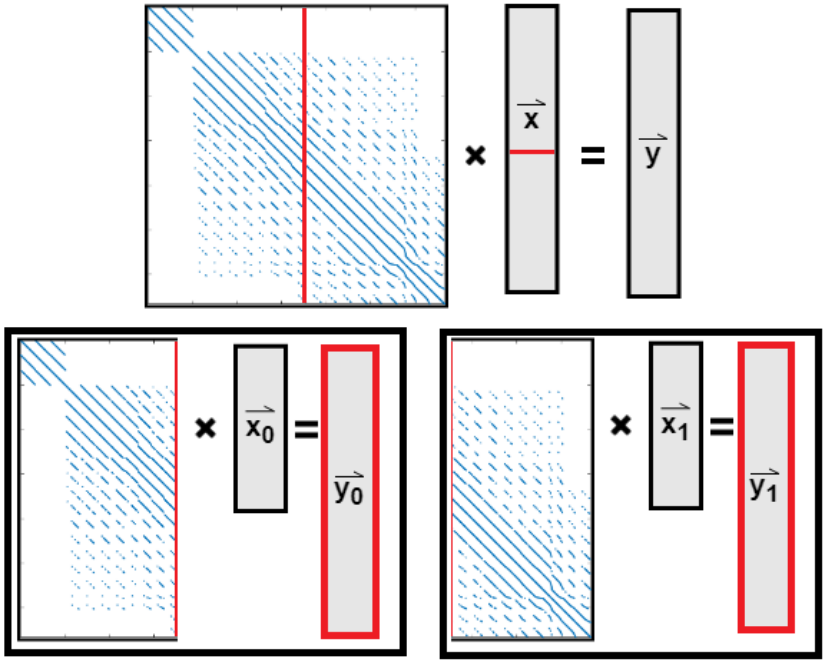
- Sparse Matrix Vector Multiplication (SpMV or SMVM).
  - Important kernel found in many **iterative** applications.
- An encoded (sparse) matrix multiplied with a vector
  - We use CSR (Compressed sparse row)

Challenge



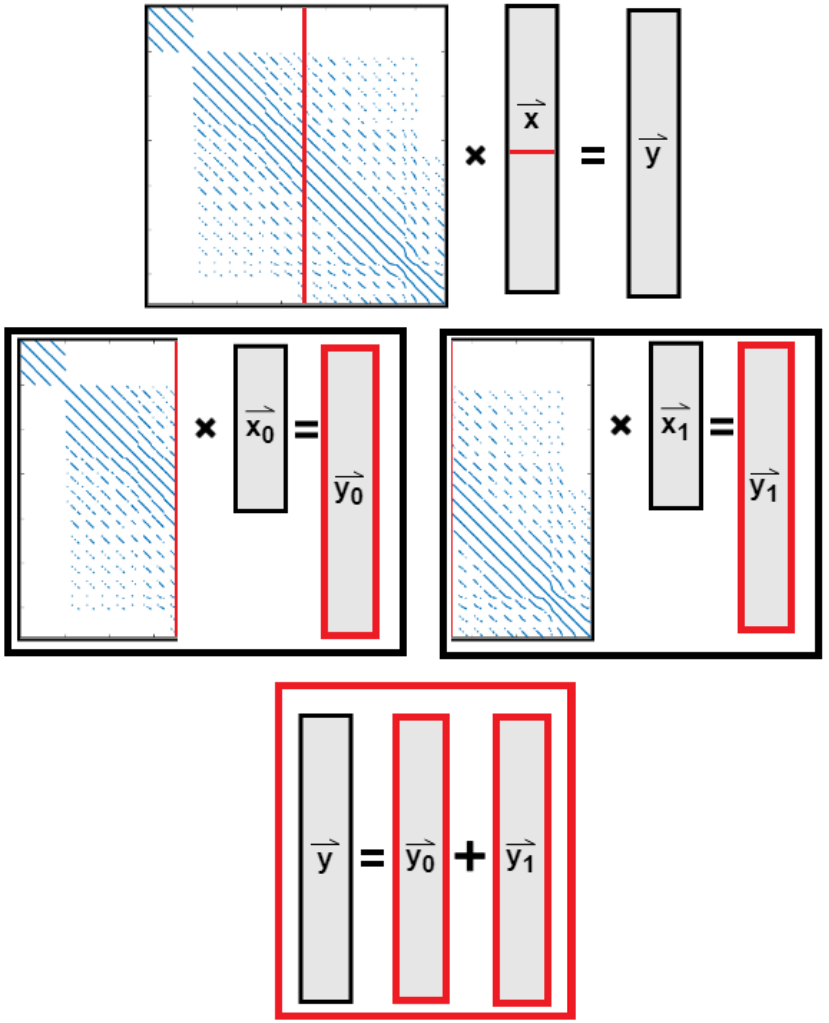
- SpMV challenging to accelerate for large-scale problems.
  - Low data locality

Challenge



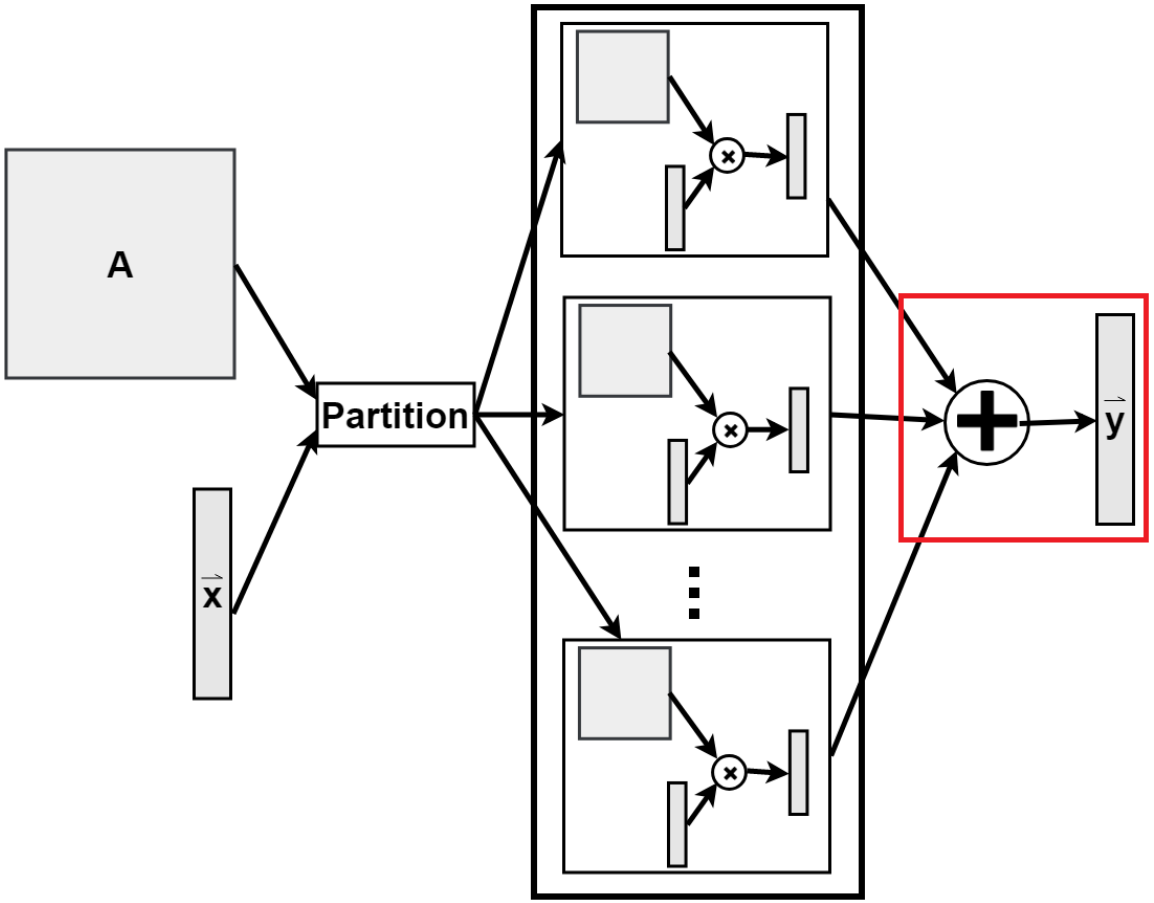
- SpMV challenging to accelerate for large-scale problems.
  - Low data locality
- Partitioning produces intermediates

Challenge



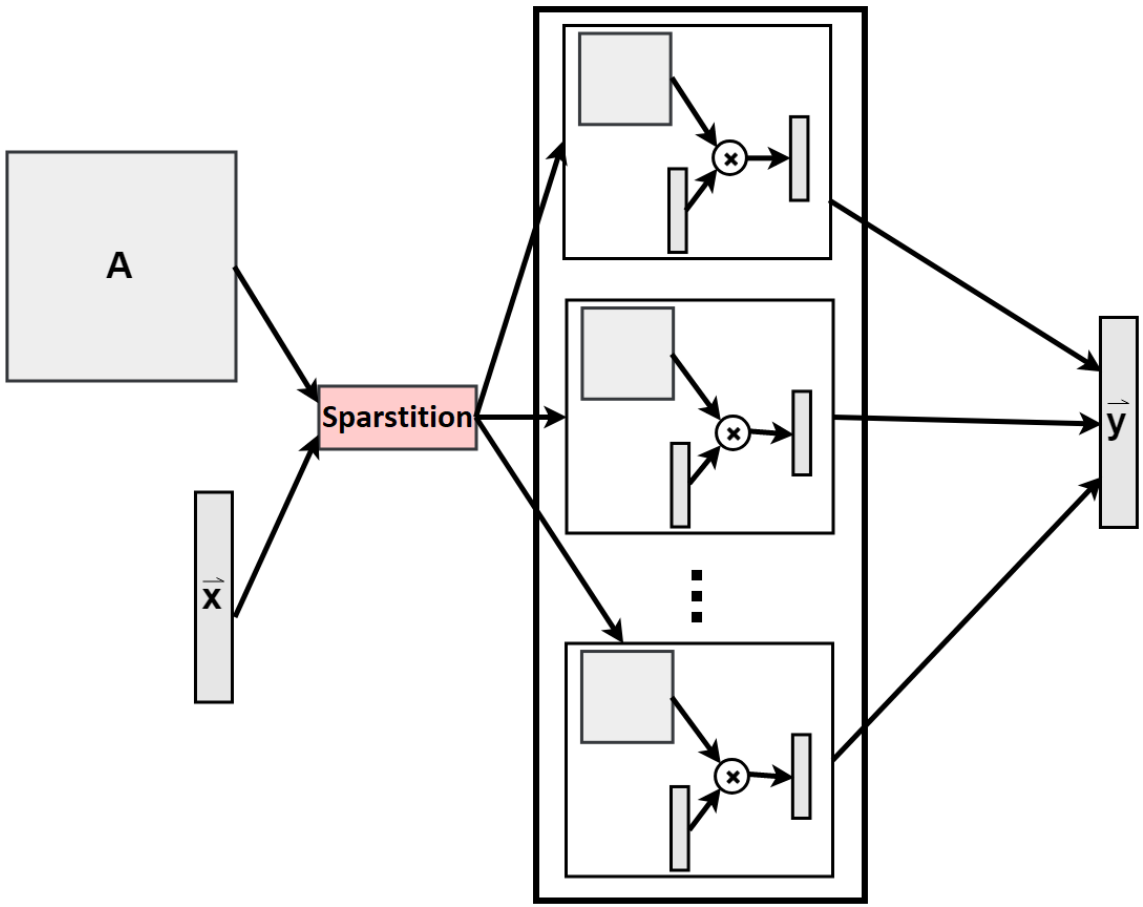
- SpMV challenging to accelerate for large-scale problems.
  - Low data locality
- Partitioning produces intermediates
- Need for merging.

Consequent Parallelism



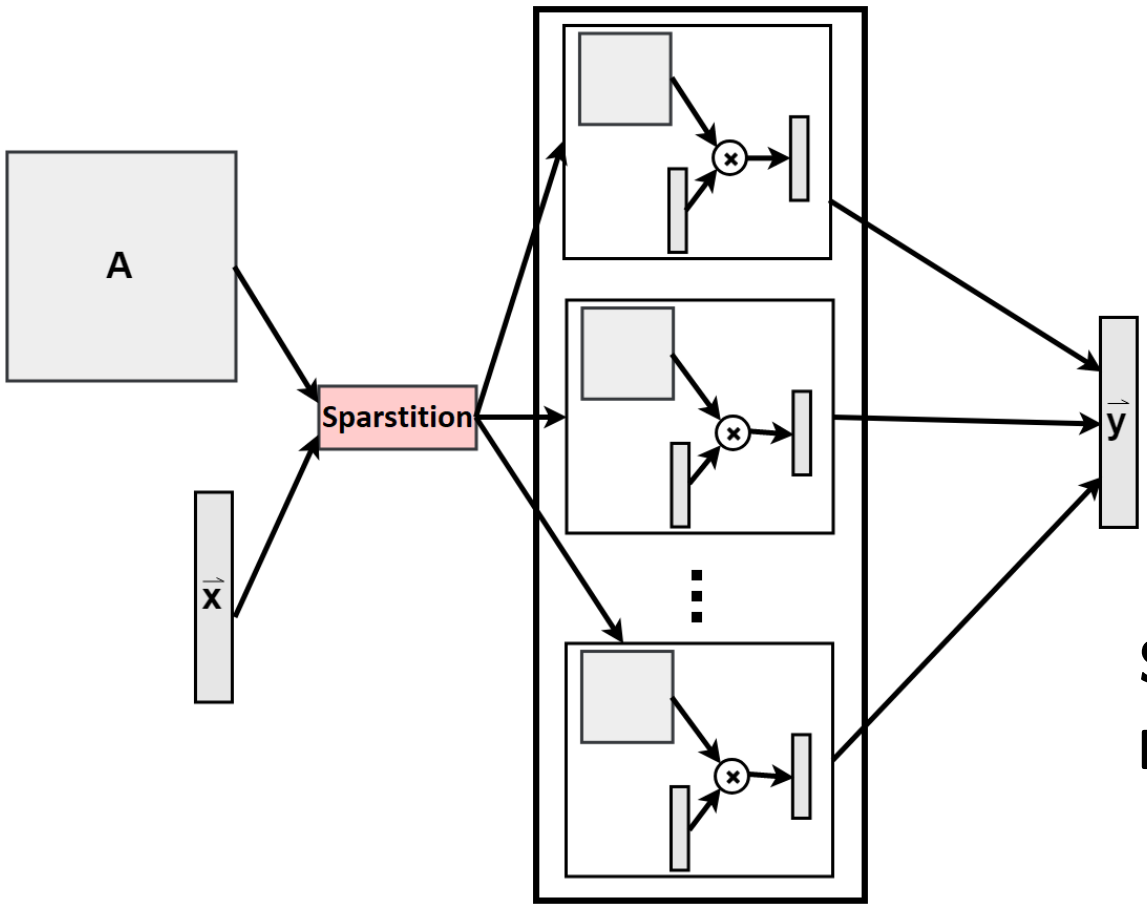
- Prohibits parallelism.

Consequent Parallelism



- Sparstition prevents generation of intermediate vectors.
  - Parallelism between partitions gained.

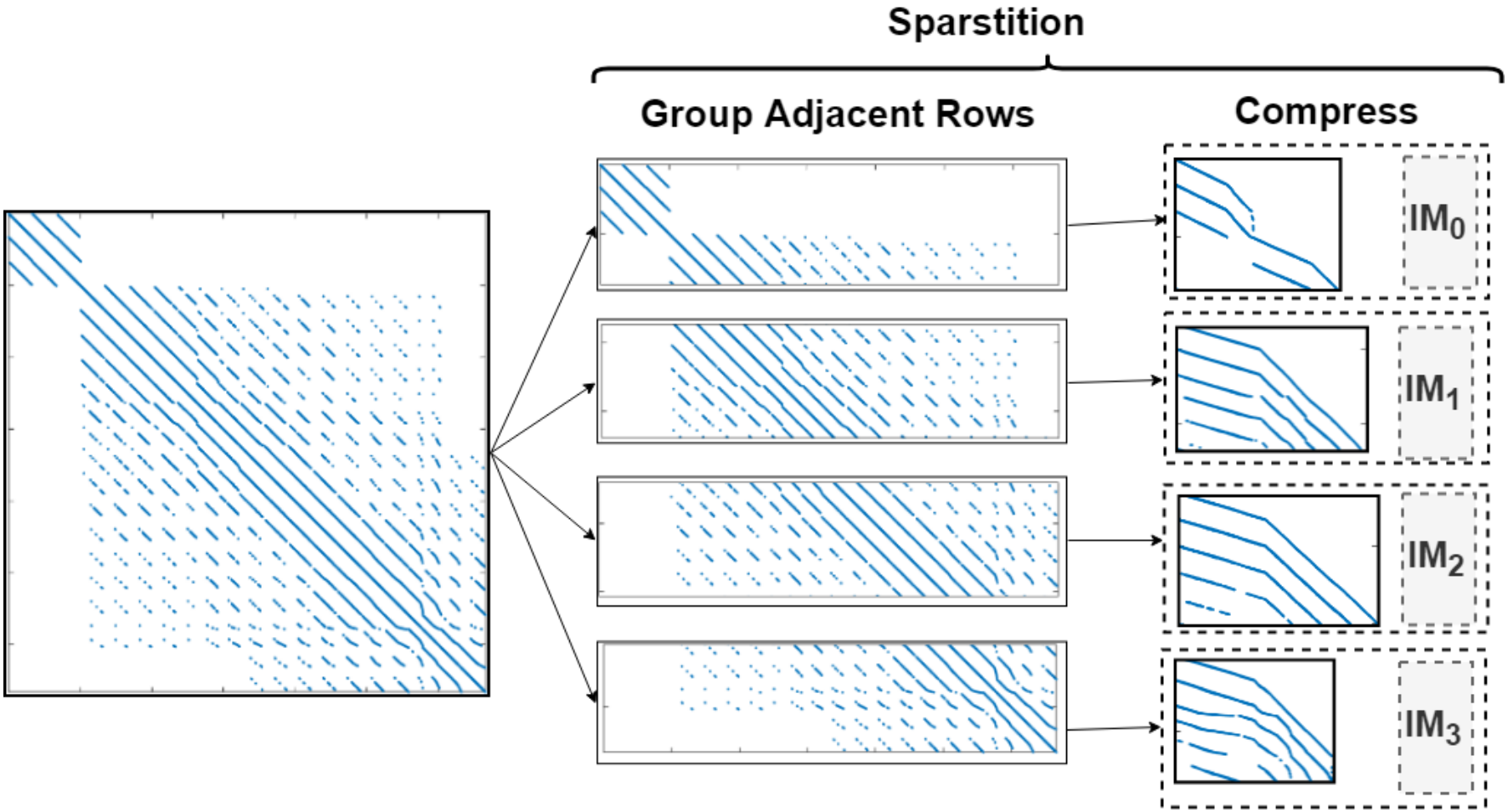
Consequent Parallelism



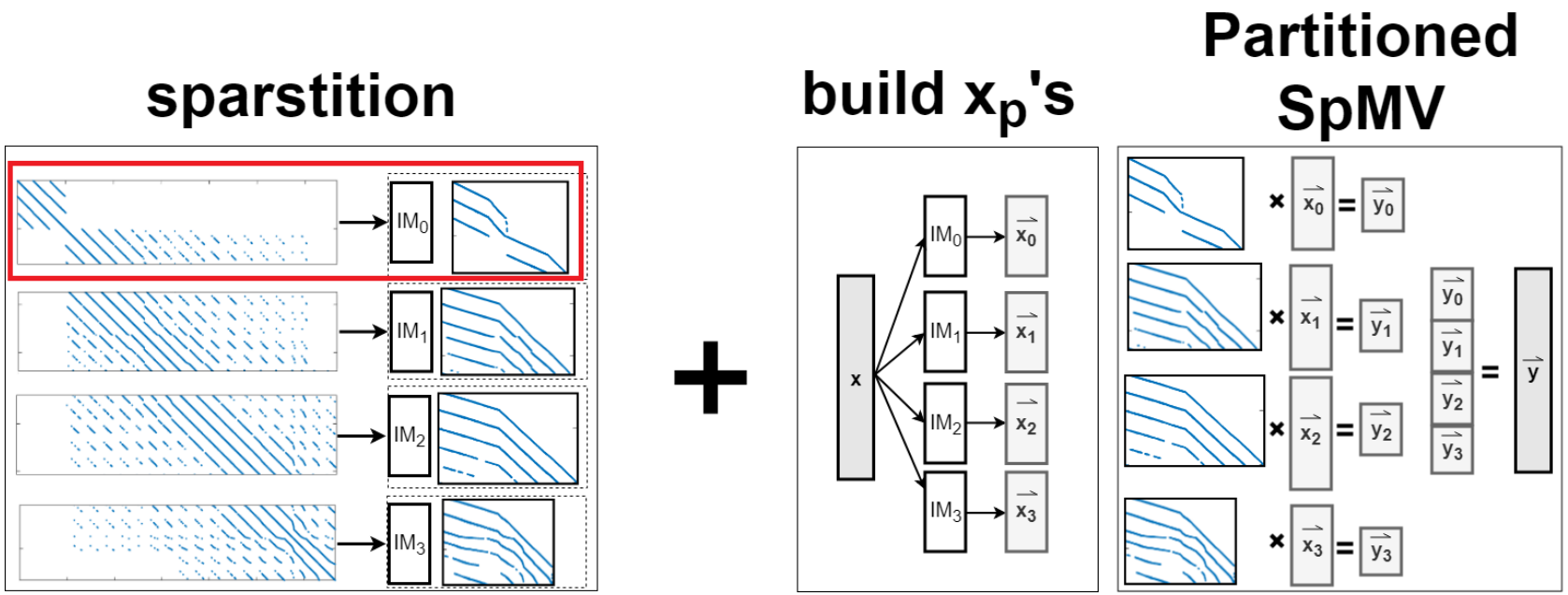
- Sparstition prevents generation of intermediate vectors.
  - Parallelism between partitions gained.

Speedup with parallel pipelines?

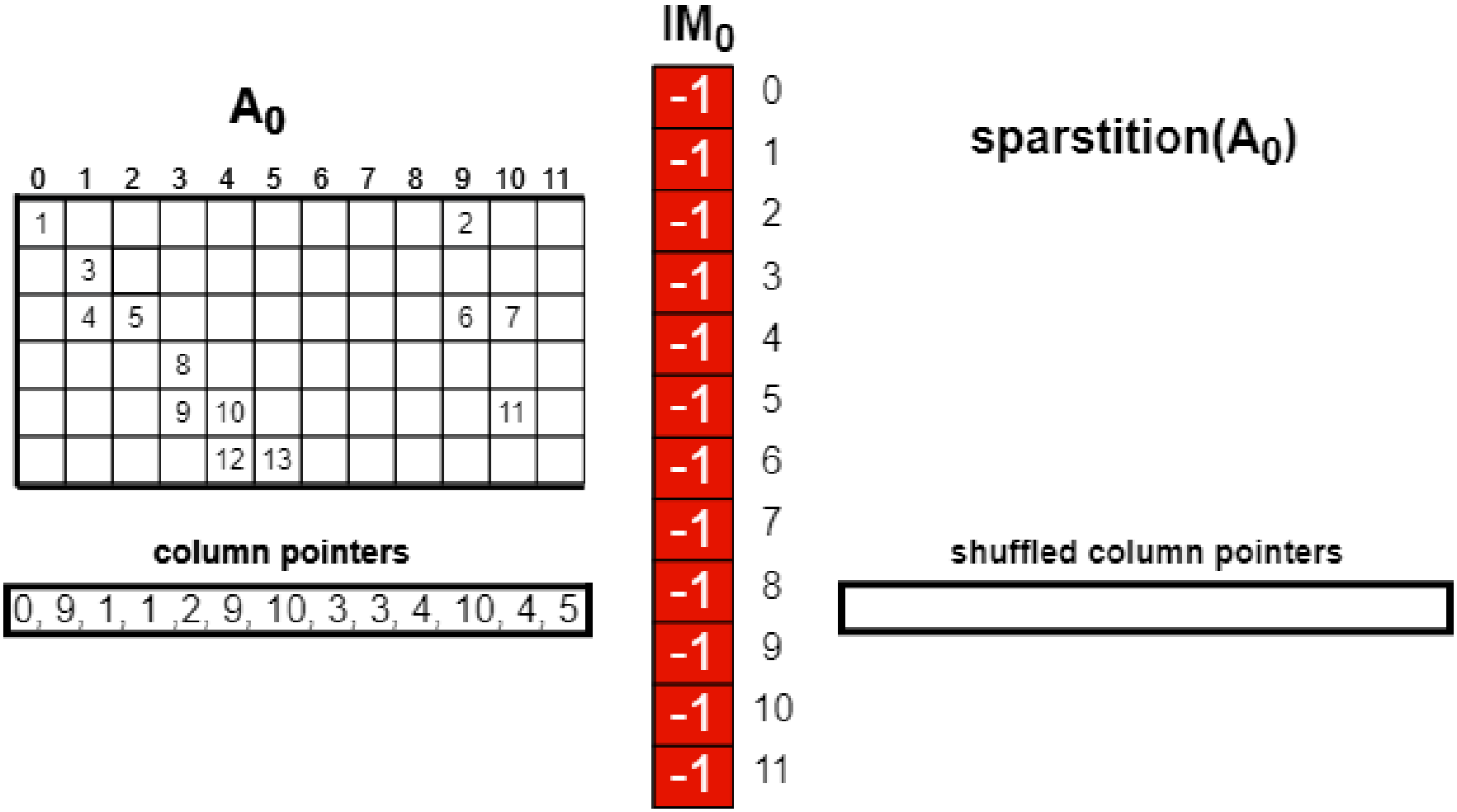
The Solution



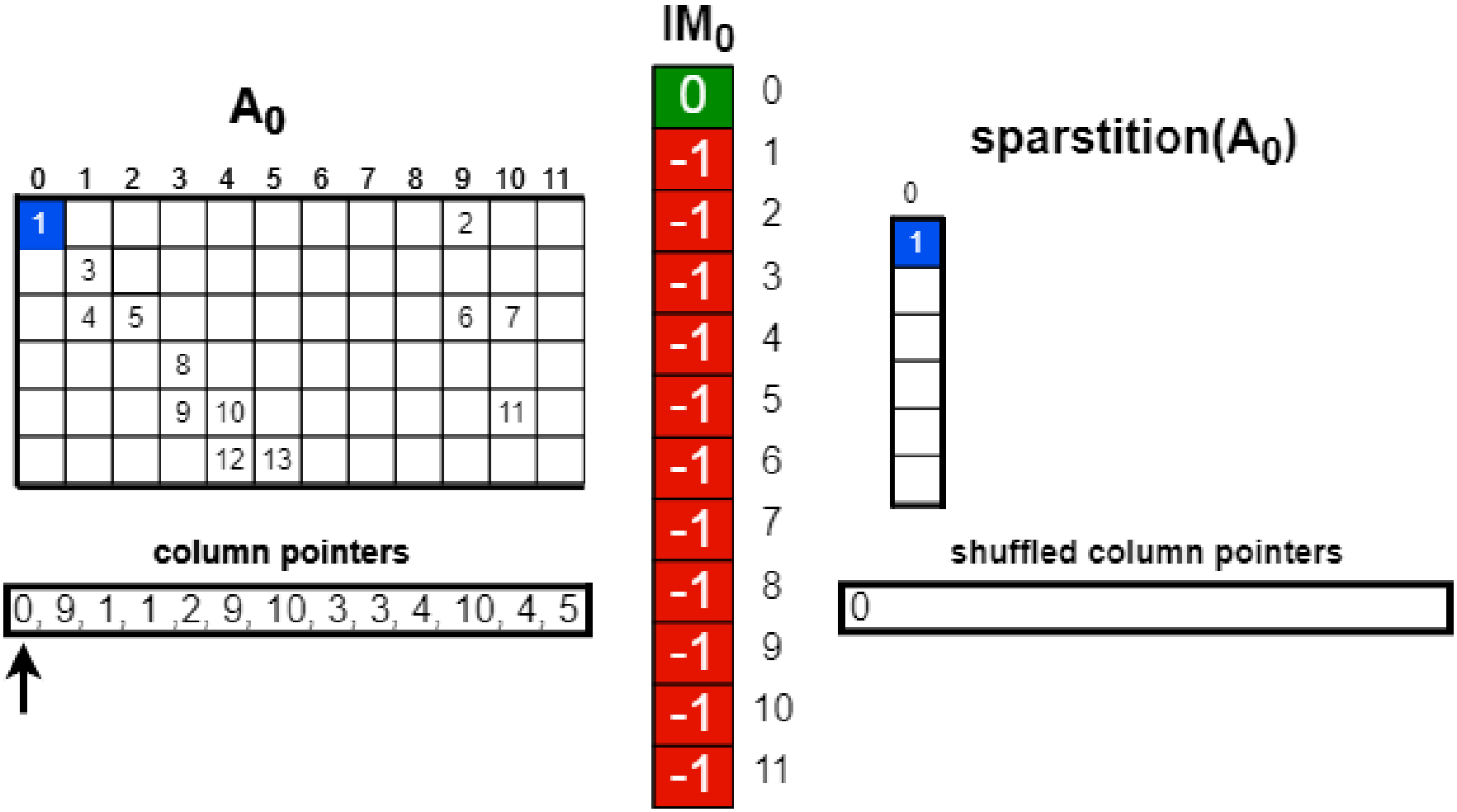
The Algorithm



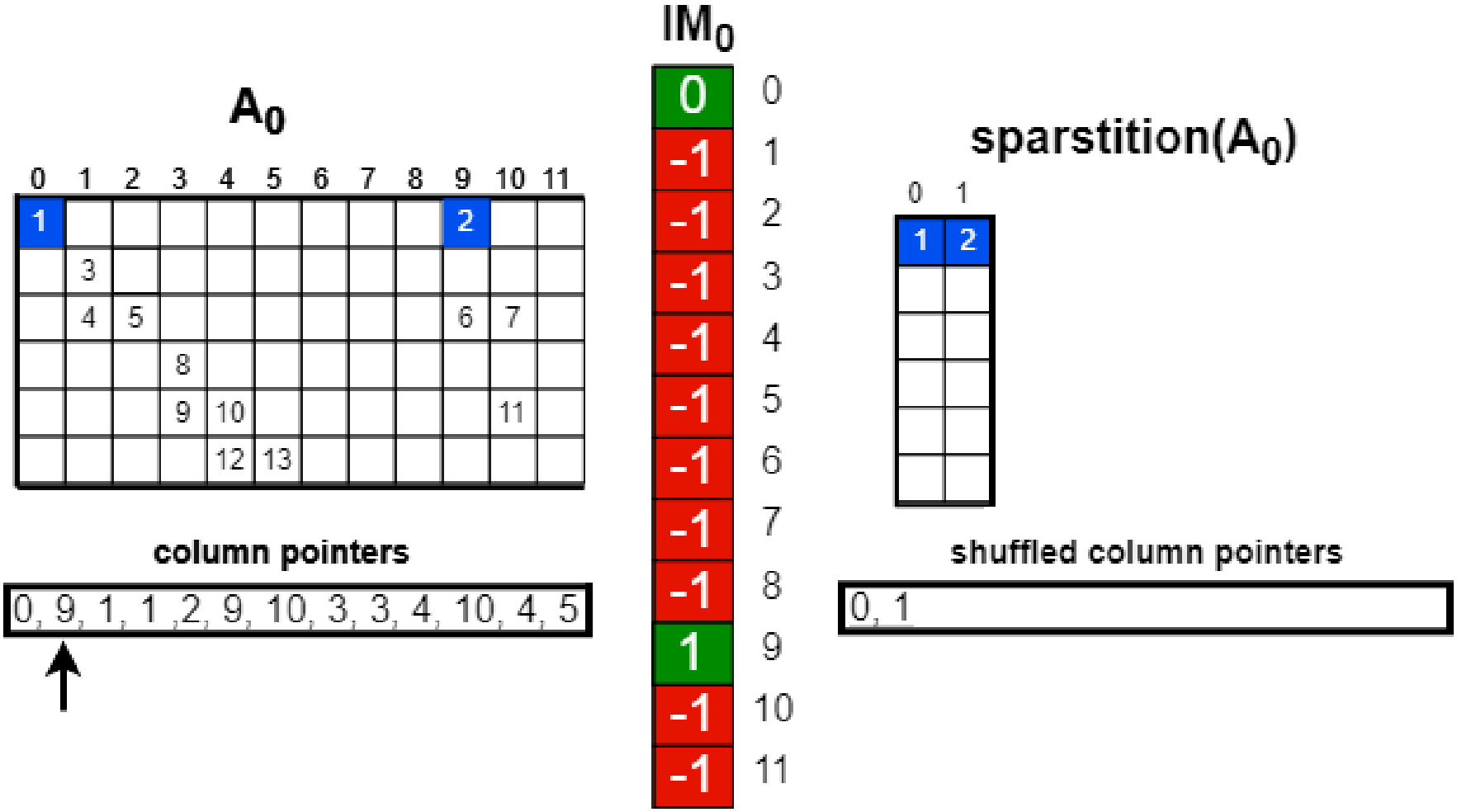
The Algorithm



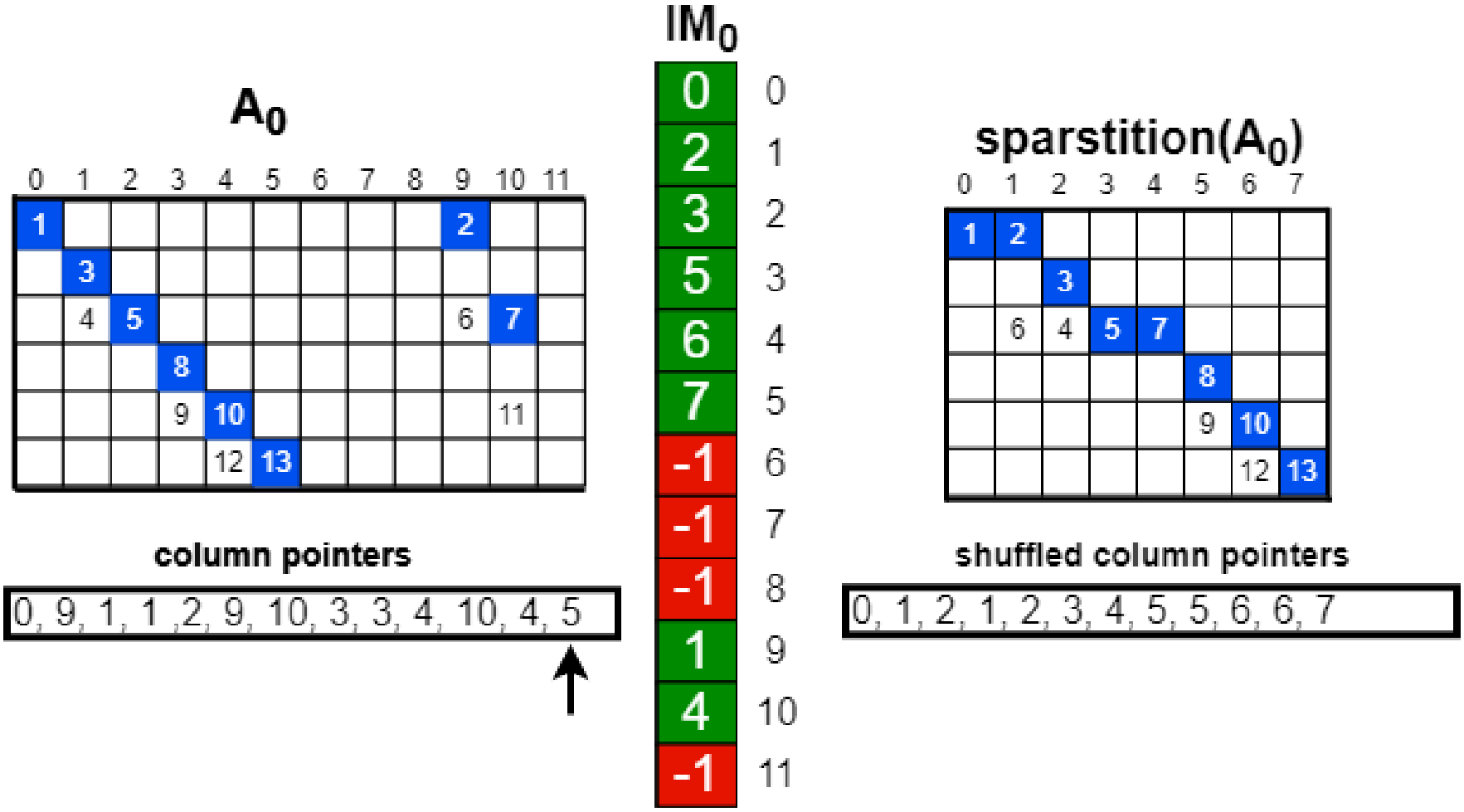
The Algorithm



The Algorithm

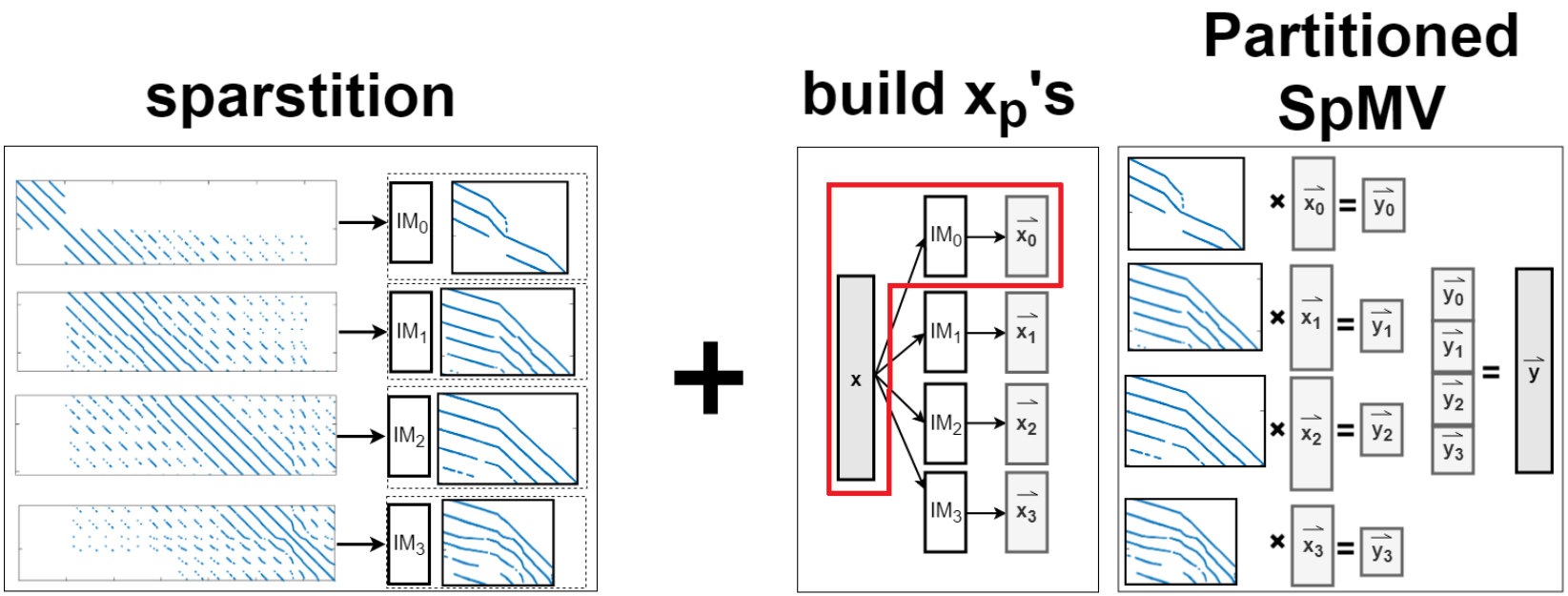


The Algorithm

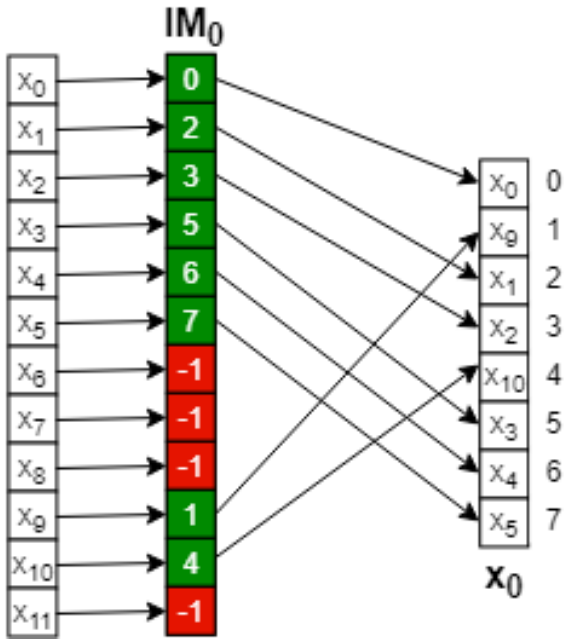


$\vec{x}$ size reduced from 12 to 8.

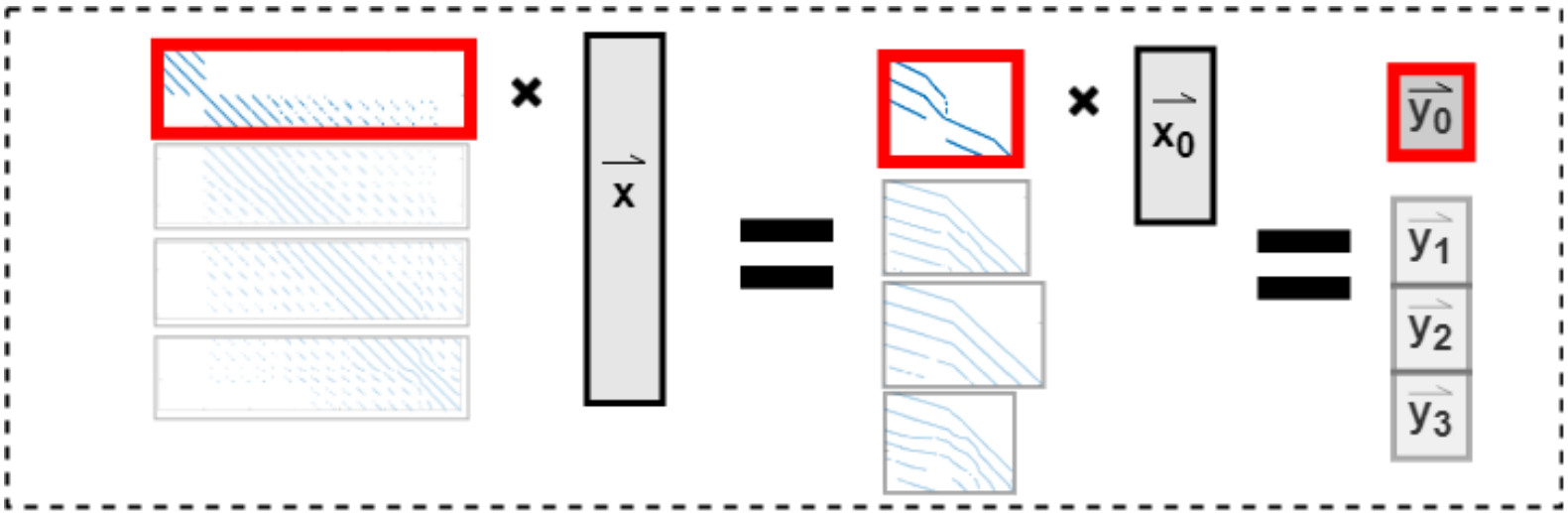
Build x_p and Compute Partition



Build x_p and Compute Partition



Build x_p and Compute Partition



Pros & Cons

- Simple
- Increases data locality
  - Eliminates risk of cache misses
  - Enables parallel processing of partitions

Pros & Cons

- Simple
- Increases data locality
  - Eliminates risk of cache misses
  - Enables parallel processing of partitions
- Reduces $\vec{x}$ size so it may fit in cache

Pros & Cons

- Simple
- Increases data locality
  - Eliminates risk of cache misses
  - Enables parallel processing of partitions
- Reduces $\vec{x}$ size so it may fit in cache

BUT

- Not suitable for all sparsity patterns
  - Ineffective for dense rows

Pros & Cons

- Simple
- Increases data locality
  - Eliminates risk of cache misses
  - Enables parallel processing of partitions
- Reduces $\vec{x}$ size so it may fit in cache

BUT

- Not suitable for all sparsity patterns
  - Ineffective for dense rows
- Values are replicated

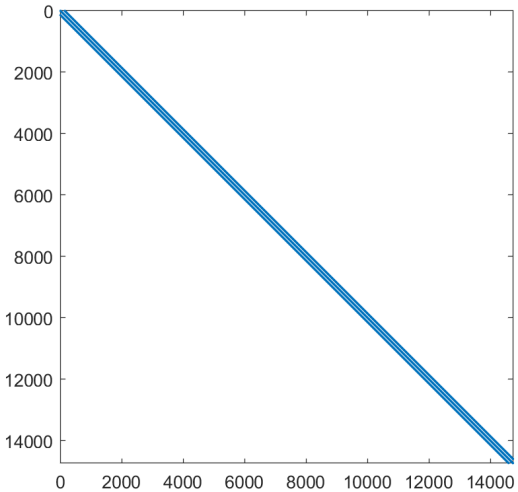
Pros & Cons

- Simple
- Increases data locality
  - Eliminates risk of cache misses
  - Enables parallel processing of partitions
- Reduces $\vec{x}$ size so it may fit in cache

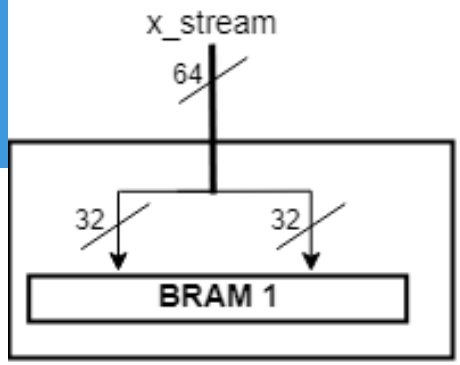
BUT

- Not suitable for all sparsity patterns
  - Ineffective for dense rows
- Values are replicated

Most effective for banded matrices

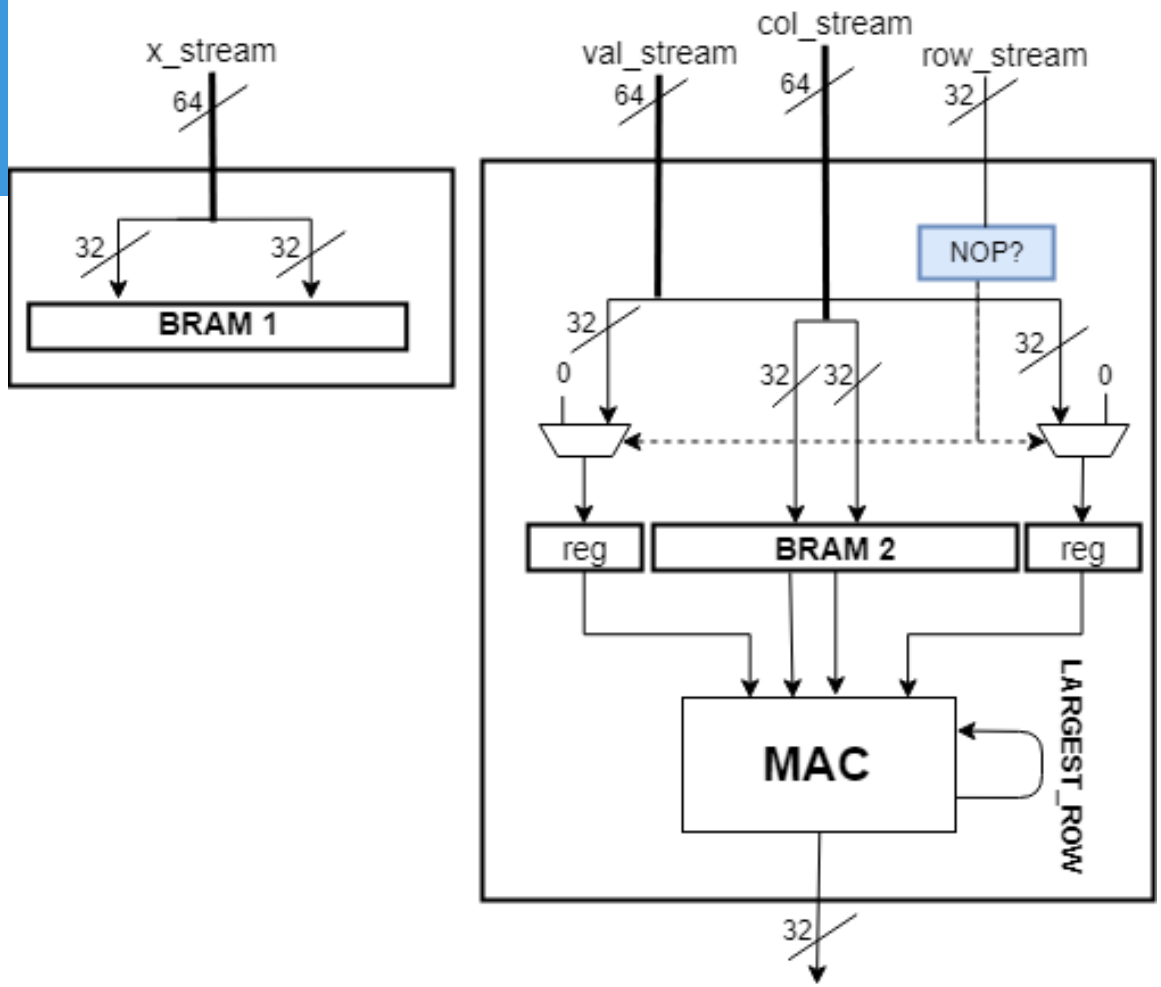


HLS Pipeline



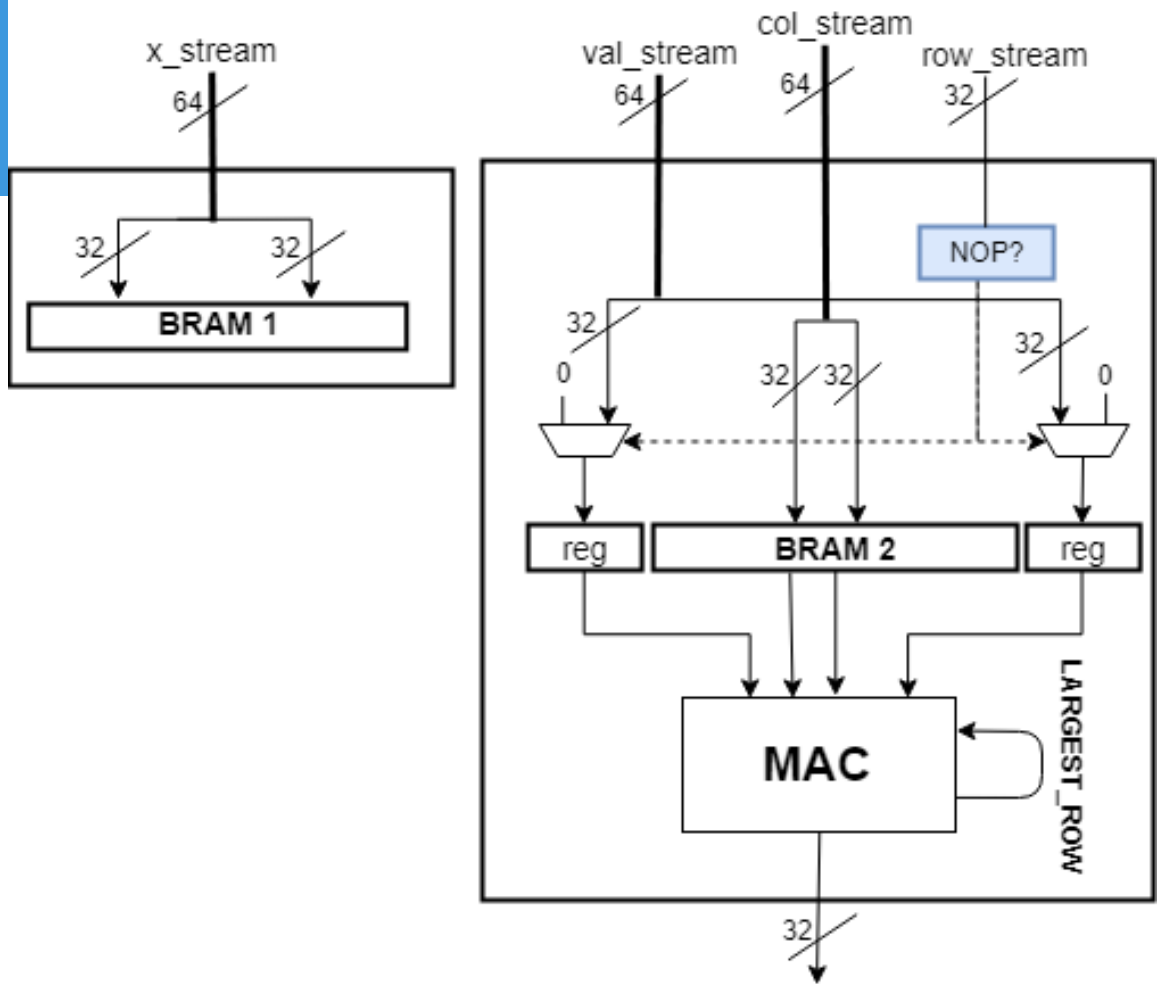
- Target: ZedBoard
- $\vec{x_p}$ is written to one of the buffers

HLS Pipeline



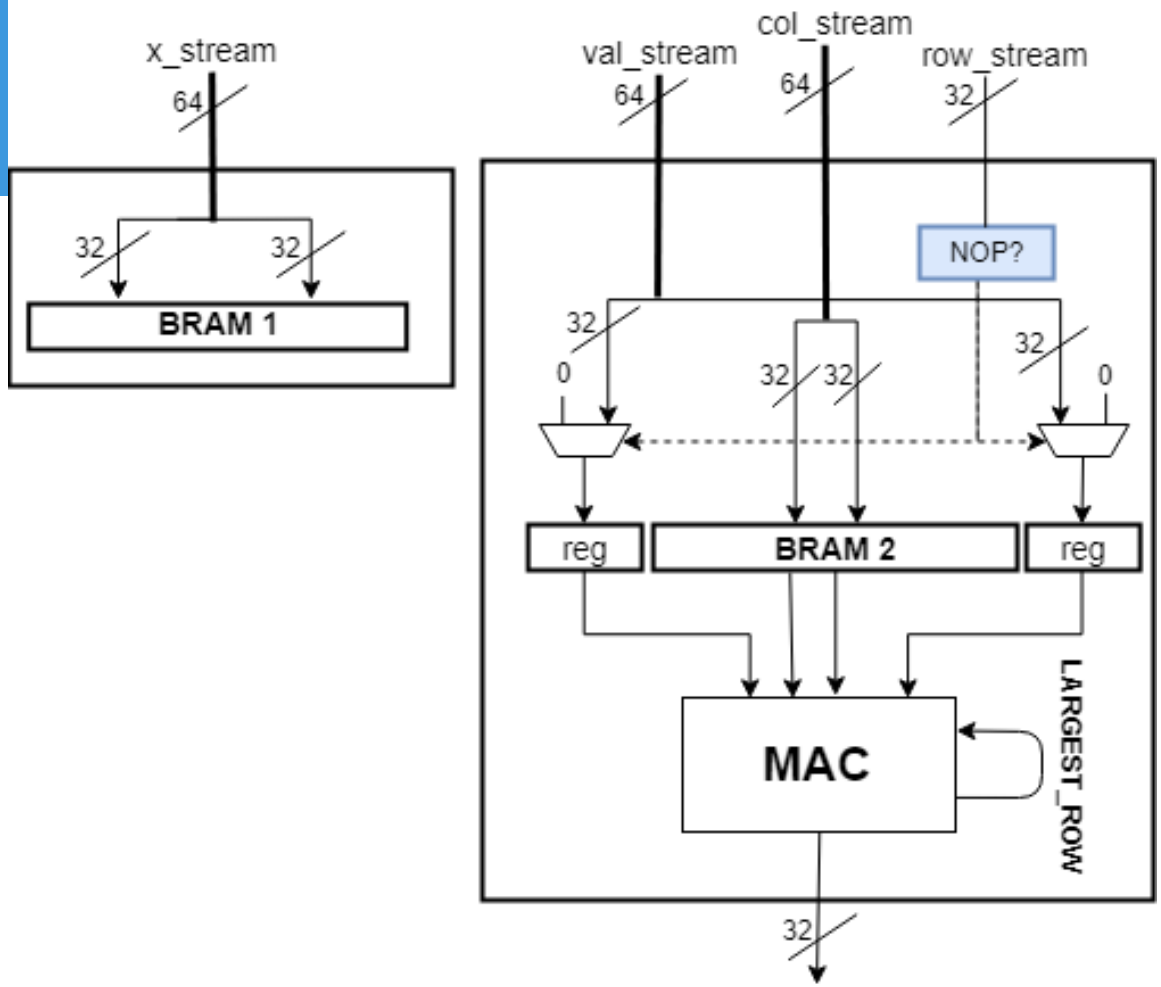
- Target: ZedBoard
- $\vec{x_p}$ is written to one of the buffers
- Meanwhile, the previous $\vec{x_p}$ is read.

HLS Pipeline



- Target: ZedBoard
- $\vec{x_p}$ is written to one of the buffers
- Meanwhile, the previous $\vec{x_p}$ is read.
- Two simultaneous Multiply-Accumulate (MAC) operations.
  - Limited by the bandwidth.

HLS Pipeline



- Target: ZedBoard
- $\vec{x_p}$ is written to one of the buffers
- Meanwhile, the previous $\vec{x_p}$ is read.
- Two simultaneous Multiply-Accumulate (MAC) operations.
  - Limited by the bandwidth.
- NOPs due to static scheduling.

HLS Kernel Performance

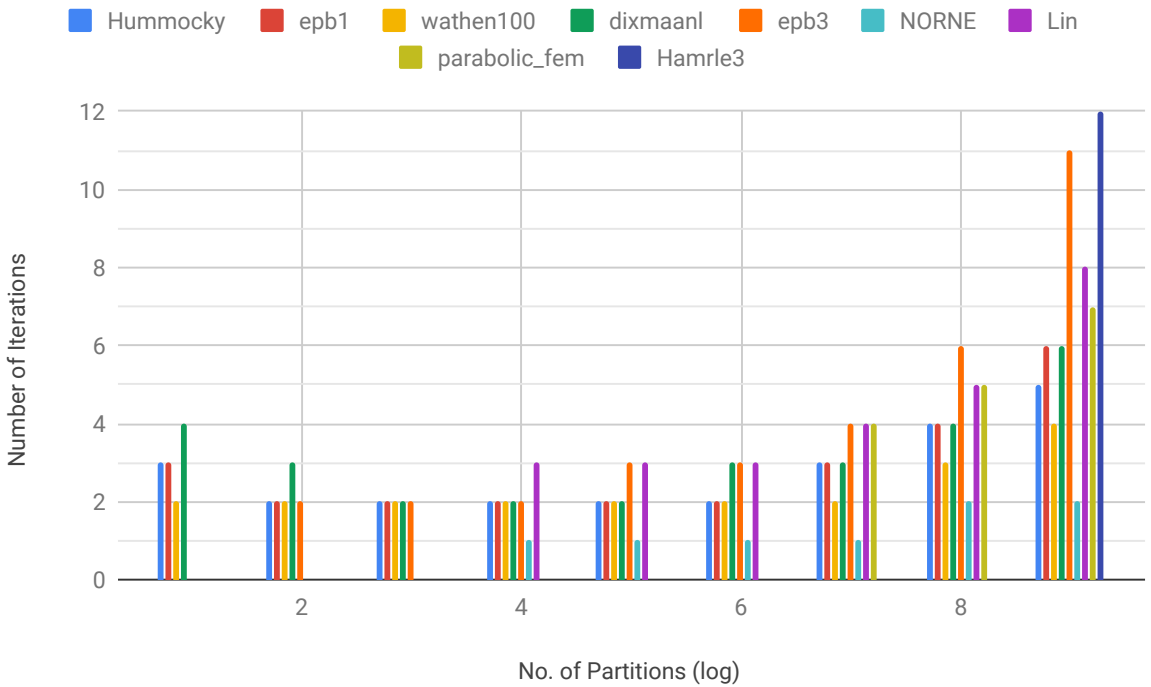
Matrix		Execution Time (ms)		
Name	**Largest row**	**[1]**	**This work**	**Speedup**
bcsstm25	6	2.8	**2.15**	1.3
dw8192	8	3.5	**0.77**	4.6
bcsstk12	27	1.0	**0.46**	2.2
ex7	75	3.0	**1.29**	2.3

- State-of-the-art (Vivado) HLS Design for SpMV
  - Results in [1] from simulation.

Sparstition Benchmarks

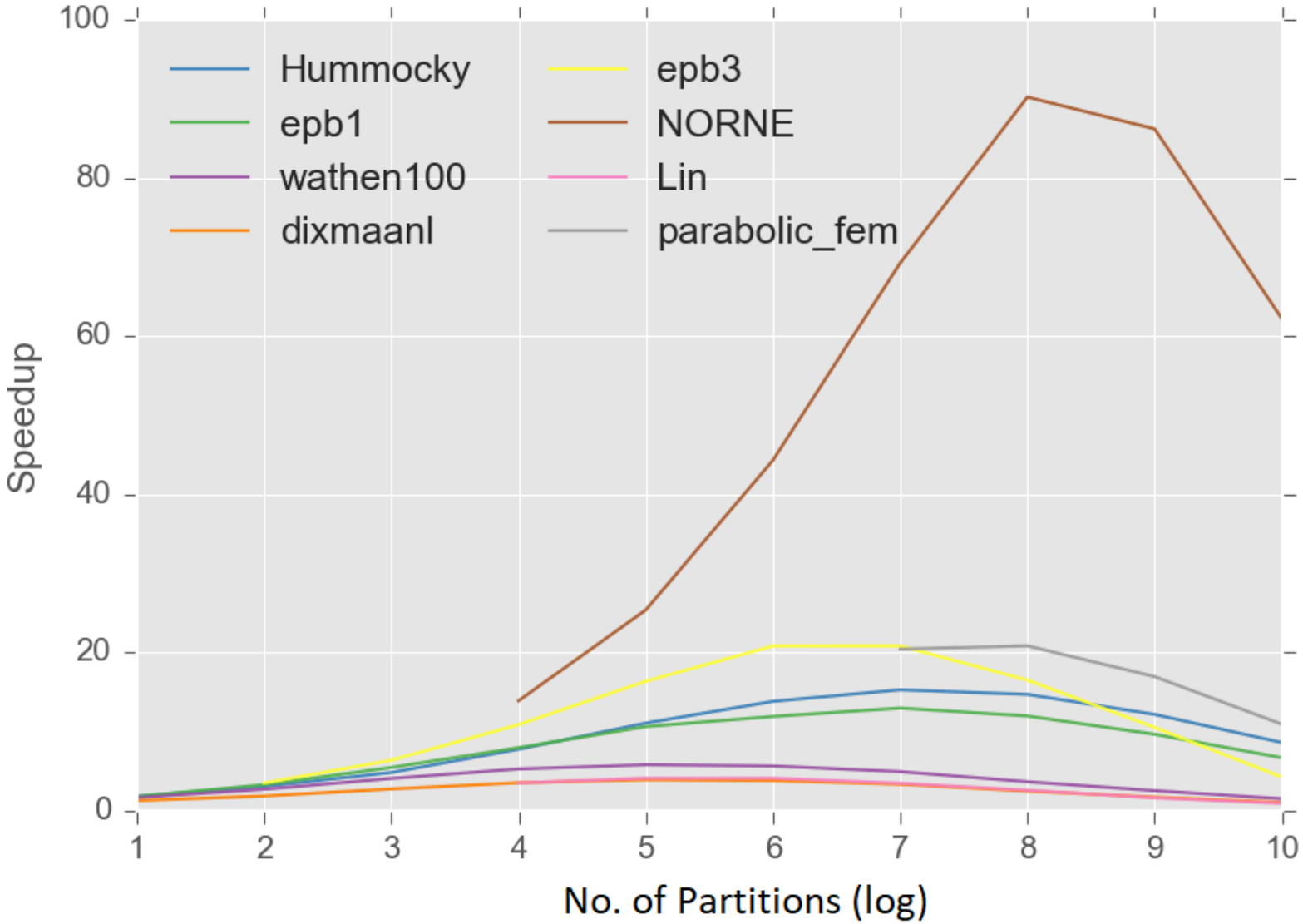
Matrix	N	NNZ	Size:Cache	Max/Avg Row	Efficiency	Application Domain
Hummocky	12,380	120,058	0.077	11 / 9.8	0.60	Oil Reservoir
epb1	14,734	95,053	0.092	7 / 6.45	0.59	Thermal Dynamics
wathen100	30,401	471,601	0.19	21 / 15.51	0.55	Random 2d/3d problem
dixmaanl	60,000	299,998	0.38	6 / 3.0	0.62	Optimization Problem
epb3	84,617	463,625	0.53	6 / 5.48	0.68	Thermal Dynamics
NORNE	133,293	2,776,851	0.83	57 / 20.83	0.29	Oil Reservoir
Lin	256,000	1,766,400	1.6	7 / 6.90	0.73	Eigenvalue problem
parabolic_fem	525,825	3,674,625	3.29	7 / 6.99	0.74	Fluid Dynamics
Hamrle3	1,447,360	5,514,242	7.23	6 / 3.81	0.57	Circuit Simulation

Theoretical Solver Speedup



Matrix	N_{l*}
Hummocky	105
epb1	90
wathen100	8
dixmaanl	9
epb3	125
NORNE	172
Lin	12
parabolic_fem	133
Hamrle3	N/A

Theoretical Solver Speedup



Conclusion

Summary

- Algorithm for large scale SpMV
- Consequent parallelism for certain sparsity patterns.
- High performance kernel for HLS standards.
- Metric to predict speedup for iterative solvers.

Conclusion

Summary

- Algorithm for large scale SpMV
- Consequent parallelism for certain sparsity patterns.
- High performance kernel for HLS standards.
- Metric to predict speedup for iterative solvers.

Future Work

- Integrate into an actual solver for large scale problems.
- Deploy with multiple parallel pipelines.

Conclusion

Summary

- Algorithm for large scale SpMV
- Consequent parallelism for certain sparsity patterns.
- High performance kernel for HLS standards.
- Metric to predict speedup for iterative solvers.

Future Work

- Integrate into an actual solver for large scale problems.
- Deploy with multiple parallel pipelines.

Thank you!


R. Garibotti, B. Reagen, Y. S. Shao, G. Wei, and D. Brooks, "Assisting high-level synthesis improve spmv benchmark through dynamic dependence analysis," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 65, no. 10, pp. 1440–1444, Oct 2018.