LU Decompositions over DAGuE

Friday Lunch Talk

Omar ZENATI, MsC Candidate

Supervisors: M. Faverge, E. Agullo, G. Bosilca, P. Ramet

May 25, 2012

Summary

LU Decomposition Algorithms

DAGuE Runtime System

Static Pivoting

A generic update engine for dynamic pivoting

Partial Pivoting

Performances

Summary

LU Decomposition Algorithms

DAGuE Runtime System

Static Pivoting

A generic update engine for dynamic pivoting

Partial Pivoting

Performances

Introduction

- The evolution (linpack, lapack, scalapack, plasma, dplasma)

Introduction

- Static algorithm:
  - Without pivoting
  - Static pivoting
  - Incremental pivoting
- Dynamic algorithm:
  - Partial pivoting
  - Total pivoting

Summary

LU Decomposition Algorithms

DAGuE Runtime System

Static Pivoting

A generic update engine for dynamic pivoting

Partial Pivoting

Performances

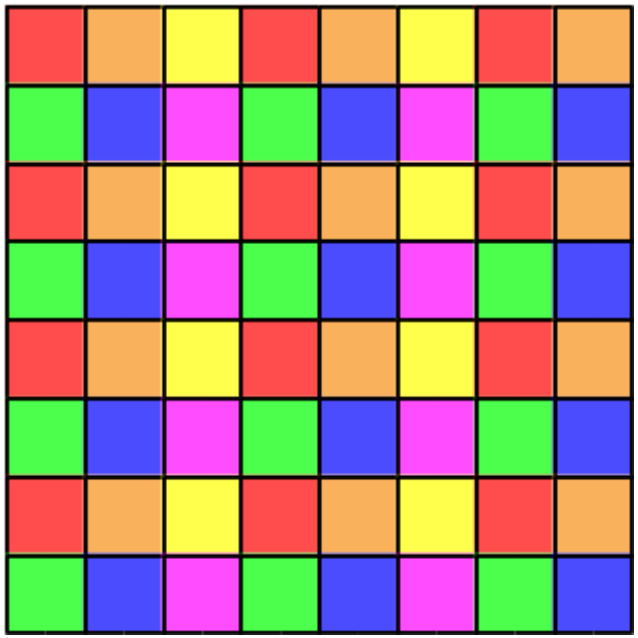
DAGuE

Quick presentation

DAGuE is a Direct Acyclic Graph scheduler Engine where :

- nodes are tasks
- edges are dependancies

Data distribution is made in 2D blocks cyclic :



DAGuE

Advantages :

- Independence between performances and computers
- Provide multicore parallelism
- Good reactivity for *load in balance*
- Natural look ahead

Problems :

- DAG is a static representation of a task flow

DAGuE

Advantages :

- Independence between performances and computers
- Provide multicore parallelism
- Good reactivity for *load in balance*
- Natural look ahead

Problems :

- DAG is a static representation of a task flow

Summary

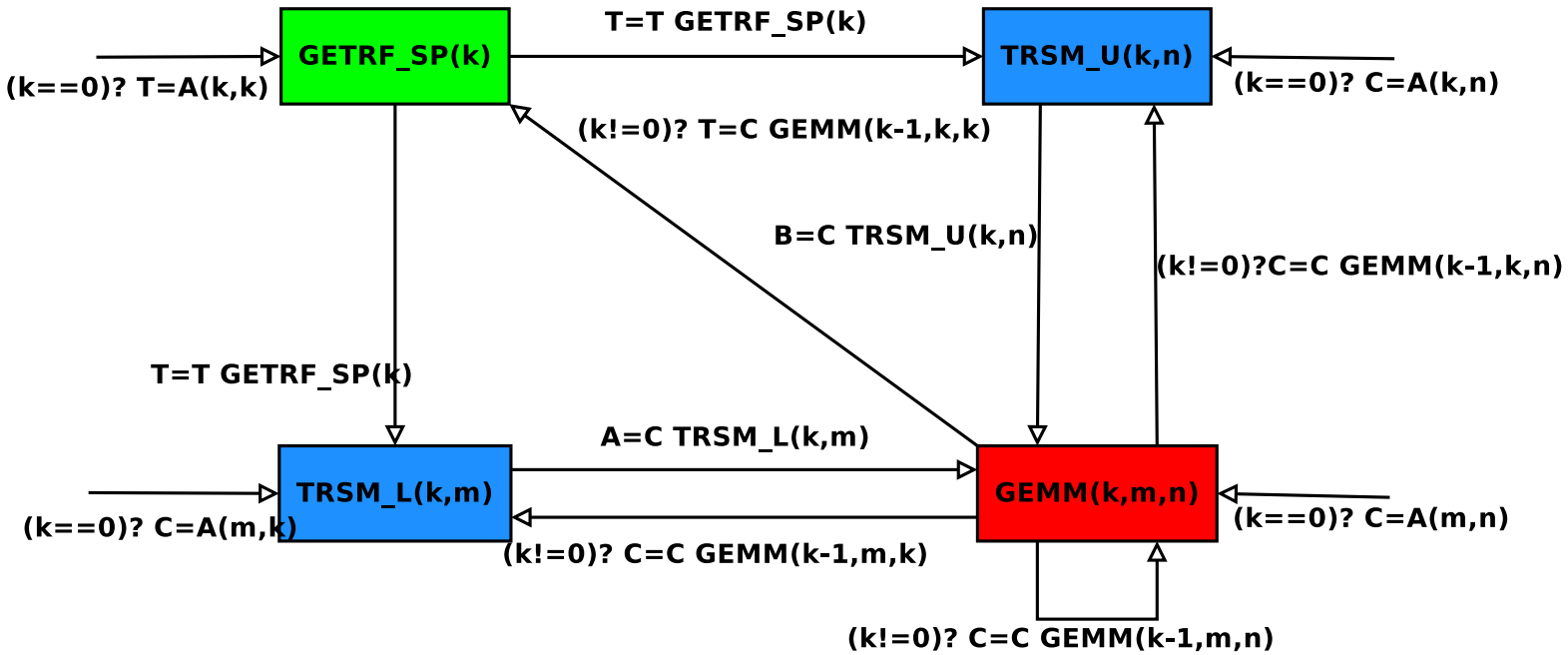
Static Pivoting

Motivation

- Static pivoting match the the task flow programming model
- Good efficiency
- Stable for several problems
- Pre treatment possible for the others
- Good upper bound for the partial pivoting

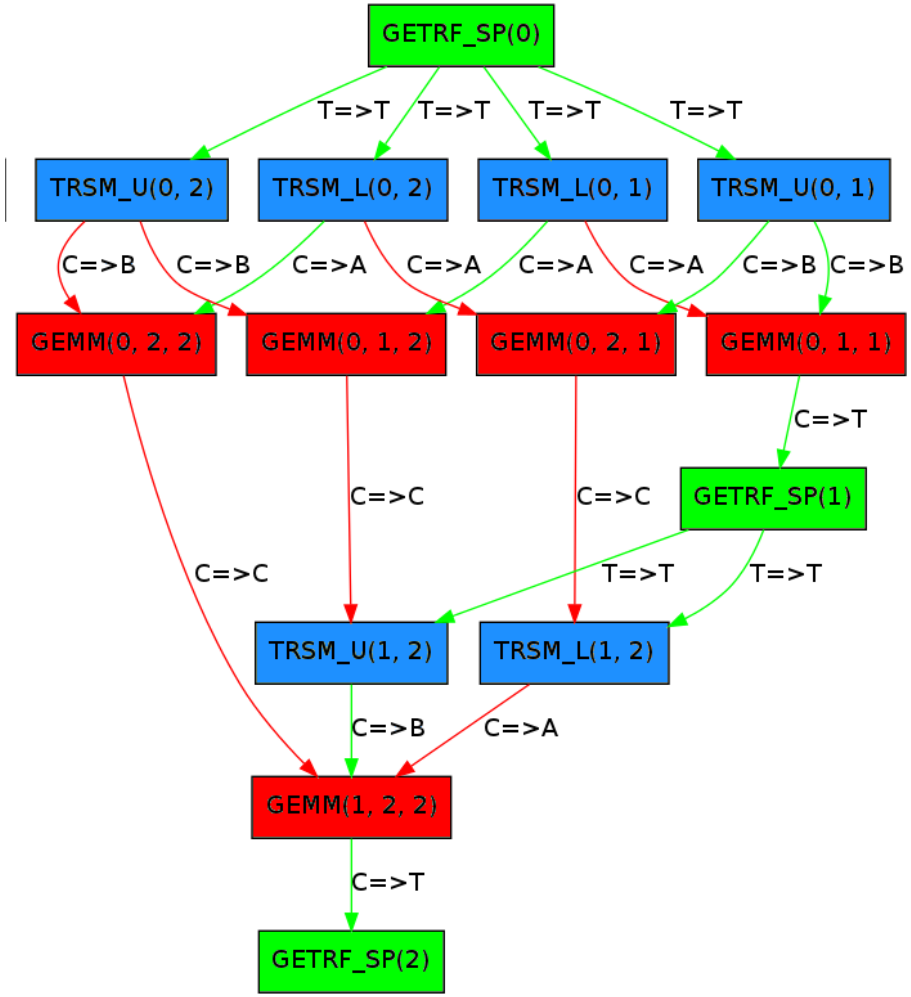
Static Pivoting

Algebraic Representation



Static Pivoting

DAG for a matrix 3*3



Summary

LU Decomposition Algorithms

DAGuE Runtime System

Static Pivoting

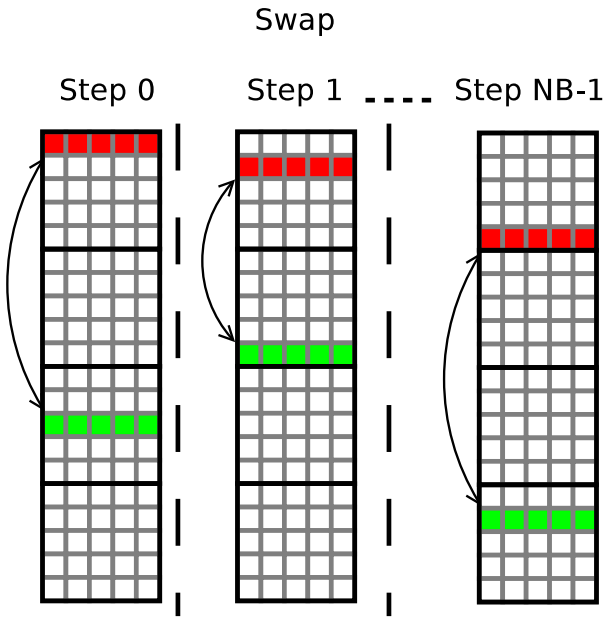
A generic update engine for dynamic pivoting

Partial Pivoting

Performances

A generic update engine for dynamic pivoting

Update Issue



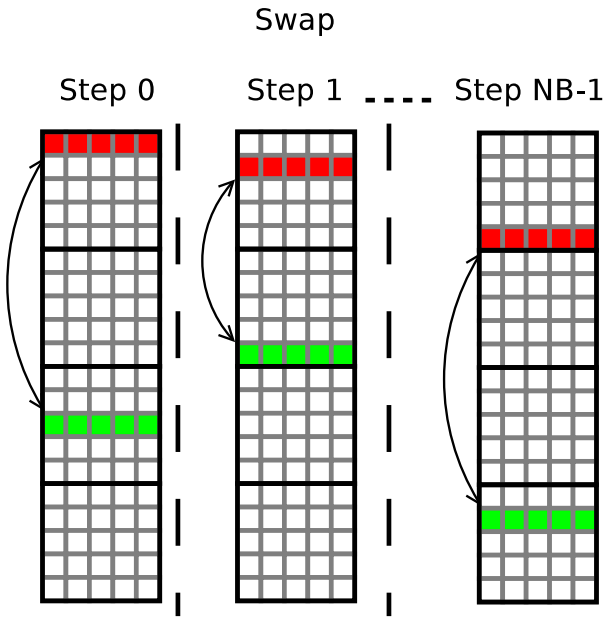
The tile U exchange swap rows with other concerned tile.

Problem

- A dynamic decision for a static DAG
  $\rightarrow$ Prepare tasks for all possible communications?

A generic update engine for dynamic pivoting

Update Issue



The tile U exchange swap rows with other concerned tile.

Problem

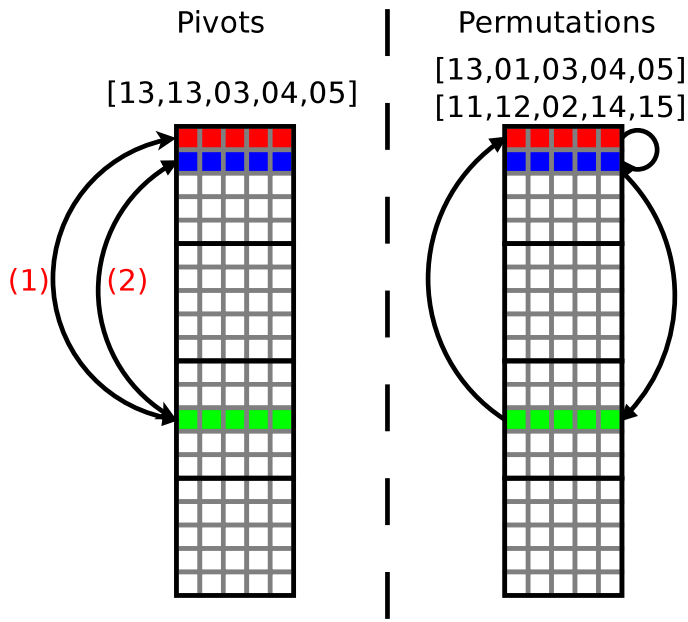
- A dynamic decision for a static DAG
  $\rightarrow$ Prepare tasks for all possible communications?

A generic update engine for dynamic pivoting

Solutions

Ideas:

- Avoiding useless swap to increase parallelism
  → Use of permutations instead of pivots indexes
- Updating the main tile is more urgent
  → Parallelize the swap **from** and the swap **into** the tile U
- Minimizing the number of communication (not the volume)
  → Gather communications of all rows over two buffers

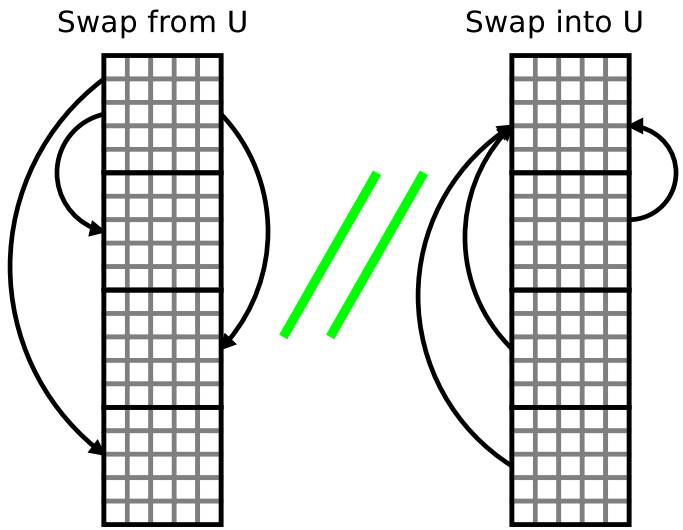


A generic update engine for dynamic pivoting

Solutions

Ideas:

- Avoiding useless swap to increase parallelism
  → Use of permutations instead of pivots indexes
- Updating the main tile is more urgent
  → Parallelize the swap **from** and the swap **into** the tile U
- Minimizing the number of communication (not the volume)
  → Gather communications of all rows over two buffers



A generic update engine for dynamic pivoting

Solutions

Ideas:

- Avoiding useless swap to increase parallelism
  $\rightarrow$ Use of permutations instead of pivots indexes
- Updating the main tile is more urgent
  $\rightarrow$ Parallelize the swap **from** and the swap **into** the tile U
- Minimizing the number of communication (not the volume)
  $\rightarrow$ Gather communications of all rows over two buffers

$\rightarrow$ Five kinds of tasks : COPY, COLLECT, RECEIVE, SEND and PASTE.

A generic update engine for dynamic pivoting

Solutions

Ideas:

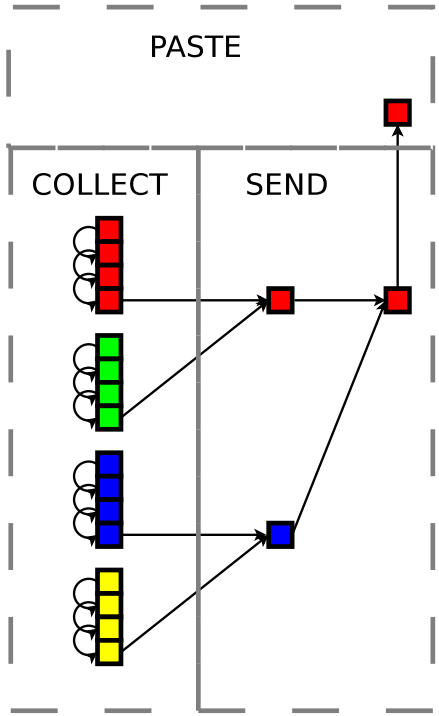
- Avoiding useless swap to increase parallelism
  $\rightarrow$ Use of permutations instead of pivots indexes
- Updating the main tile is more urgent
  $\rightarrow$ Parallelize the swap **from** and the swap **into** the tile U
- Minimizing the number of communication (not the volume)
  $\rightarrow$ Gather communications of all rows over two buffers

$\rightarrow$ Five kinds of tasks : COPY, COLLECT, RECEIVE, SEND and PASTE.

A generic update engine for dynamic pivoting

Swap into U

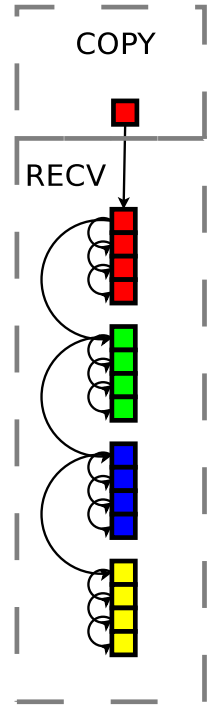
- COLLECT: Collecting the lines needed by the tile U into a buffer.
- SEND: Gather the buffers collected by COLLECT of each node.
- PASTE: Overwrite the tile U with the buffer.
- RECEIVE
- COPY



A generic update engine for dynamic pivoting

Swap from U

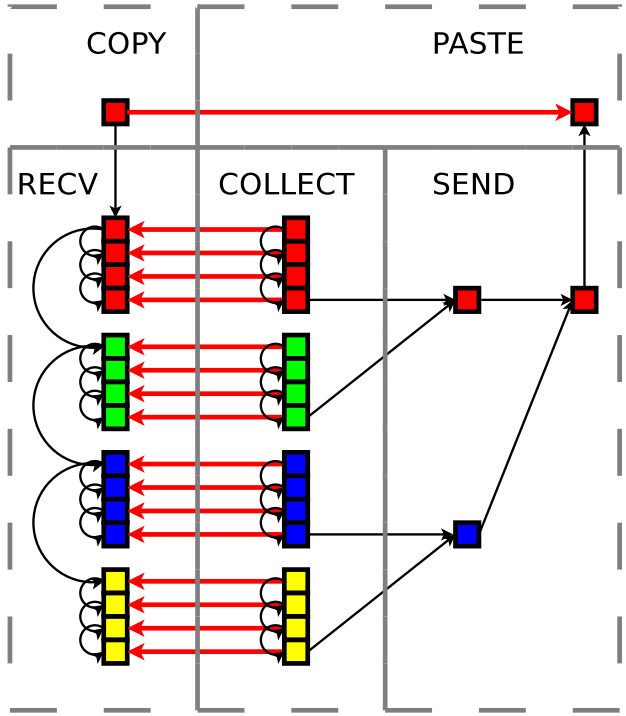
- COLLECT
- SEND
- PASTE
- COPY: Copy tile U into a buffer.
- RECEIVE: Receive the buffer U and make the swap from it.



A generic update engine for dynamic pivoting

Update Tasks Synchronisation

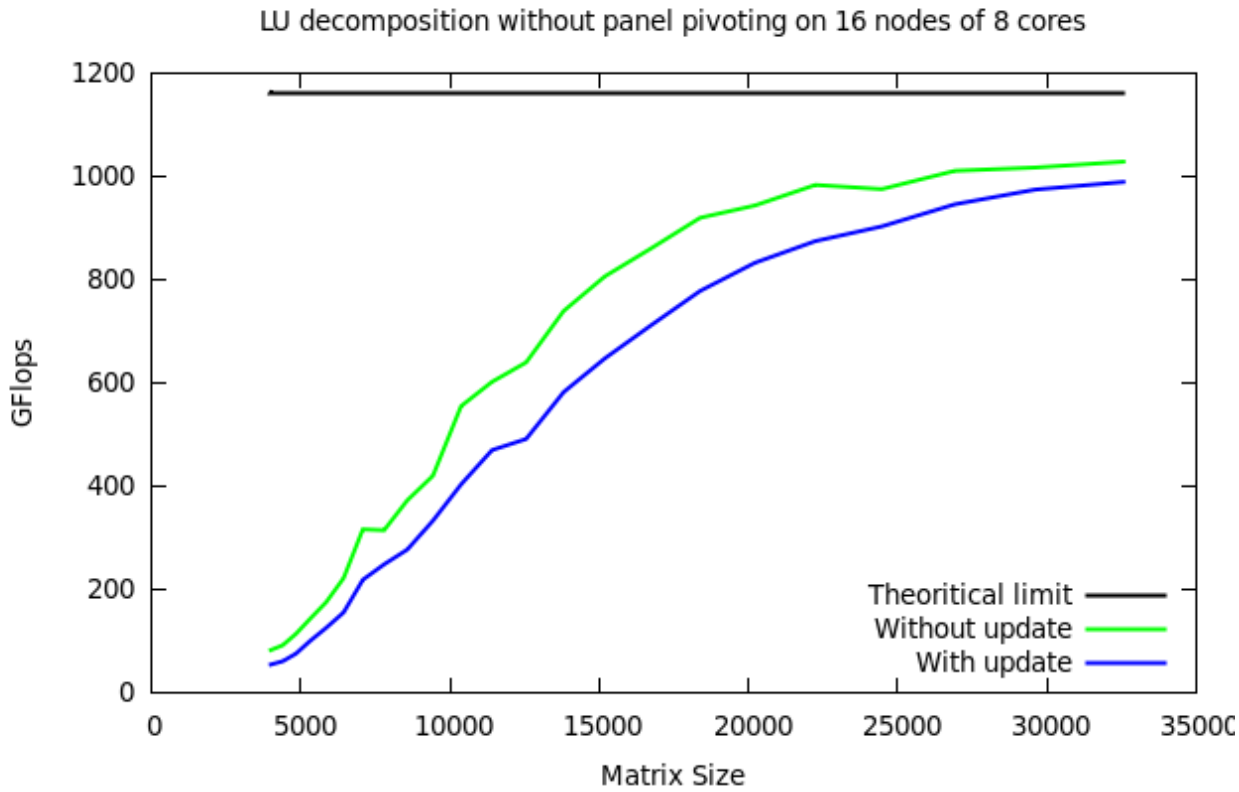
- COLLECT
- COPY
- RECEIVE
- SEND
- PASTE



The red arrows prevent the **READ AFTER WRITE**.

A generic update engine for dynamic pivoting

Update Impact



A generic update engine for dynamic pivoting

Results

- Small impact on the performance
- A generic update engine

Summary

LU Decomposition Algorithms

DAGuE Runtime System

Static Pivoting

A generic update engine for dynamic pivoting

Partial Pivoting

Performances

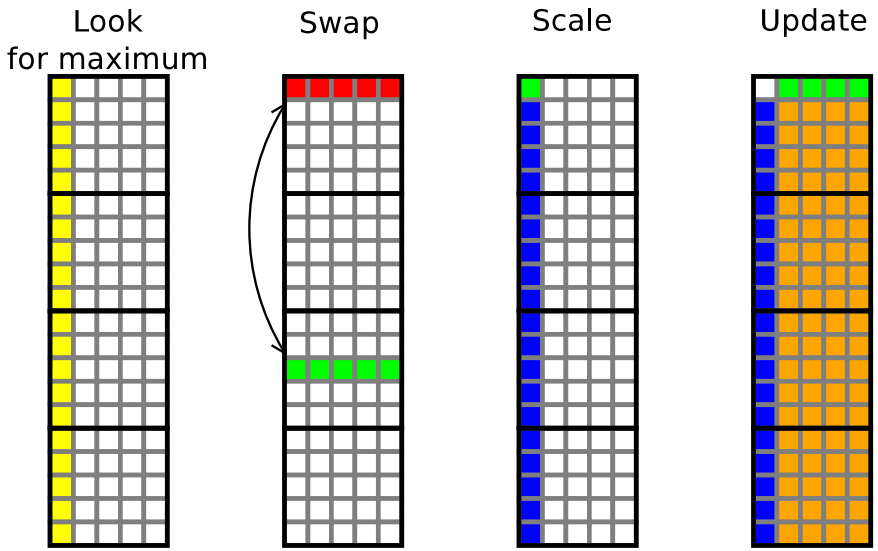
Partial Pivoting

Panel Factorization

- Several algorithms to factorize the panel:
  - LU decomposition
  - Recursive LU decomposition
  - Communication Avoiding LU
- Several way to implement:
  - threads + MPI
  - Task flow

Partial Pivoting

Operations of Panel LU Decomposition

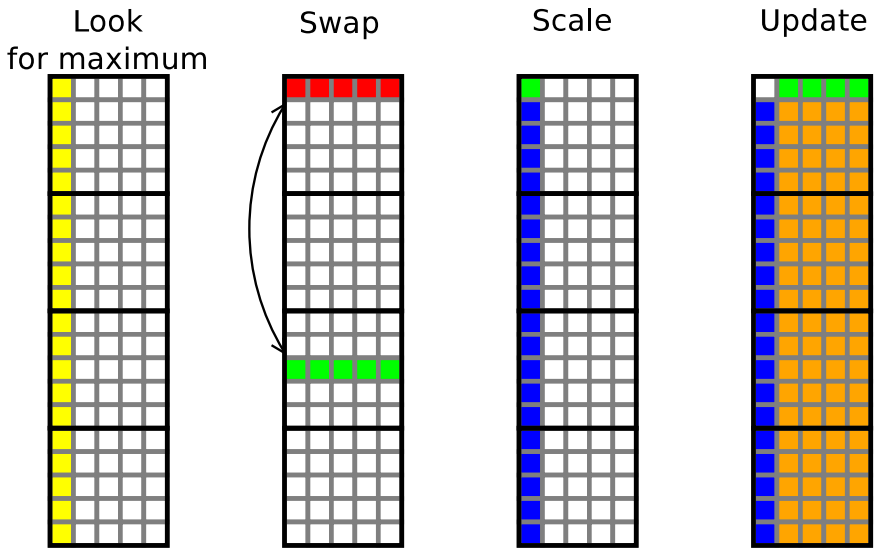


Problem for implementing with task flow

- Swap line is dynamically decided but the DAG is static
- Minimaze latency for the panel

Partial Pivoting

Operations of Panel LU Decomposition



Problem for implementing with task flow

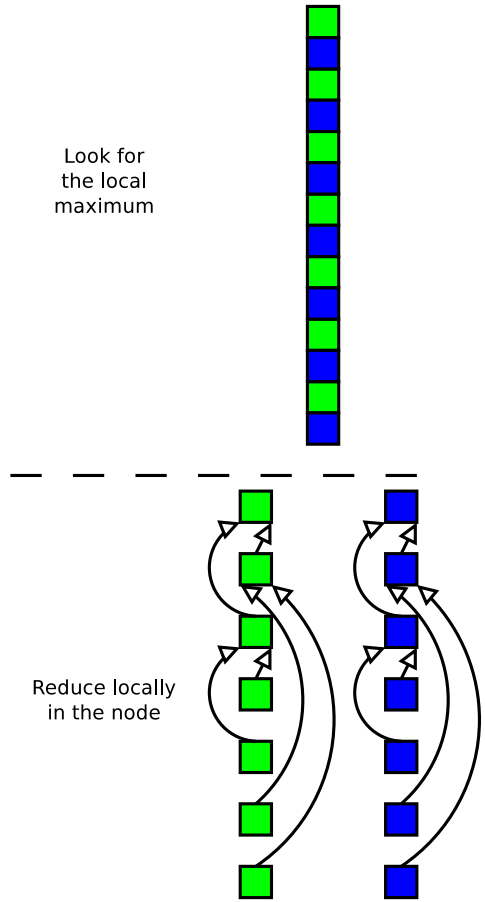
- Swap line is dynamically decided but the DAG is static
- Minimaze latency for the panel

Panel Factorization

Solutions

Solutions:

- Start for looking to the maximum localy then reduce locally the result
- Share the global result by using Bruck's algorithm
- Use internal blocking

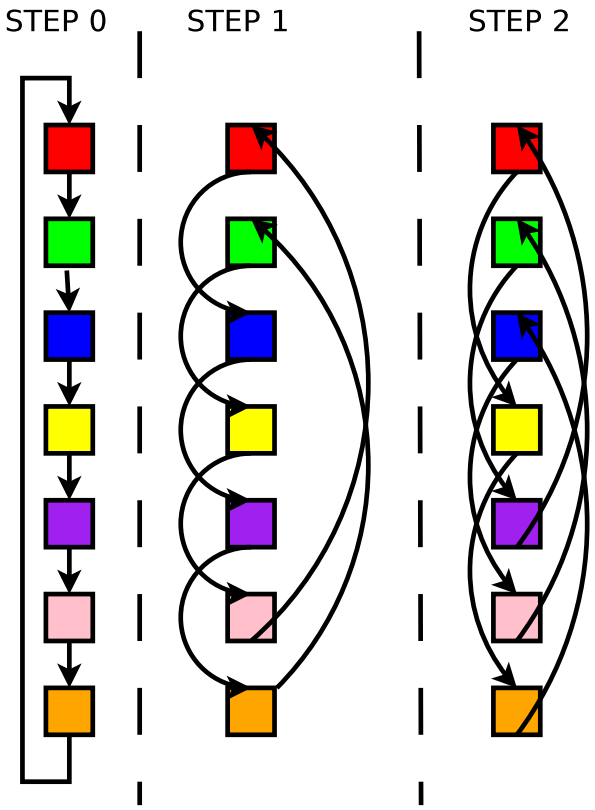


Panel Factorization

Implemented version

Solutions:

- Start for looking to the maximum localy then reduce locally the result
- Share the global result by using Bruck's algorithm
- Use internal blocking



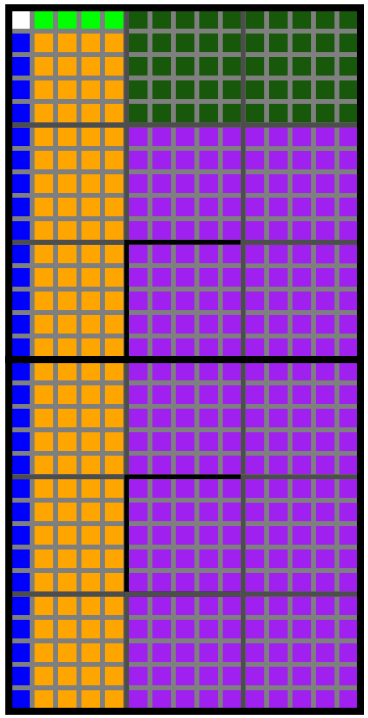
Panel Factorization

Implemented version

Optimizations:

- Start for looking to the maximum localy then reduce locally the result
- Share the global result by using Bruck's algorithm
- Use internal blocking

Panel update



Summary

LU Decomposition Algorithms

DAGuE Runtime System

Static Pivoting

A generic update engine for dynamic pivoting

Partial Pivoting

Performances

Performances of partial pivoting

- Shared memory
- Problem scalability
- Strong scalability

Conclusion and future work