

x100

# Using Domain Decomposition to strong scale past 100 GPUs

---

Michael Clark

Harvard University

Ron Babich, Rich Brower, Balint Joo,  
Steve Gottlieb and Guochun Shi



# Outline

---

- Motivation
- Domain Decomposition
- Results
- Conclusions and Outlook

# Motivation

---

- Small scale generation and analysis calculations very efficient on small clusters of GPUs
  - Capacity computing
  - E.g., Bonati *et al*, Alexandru *et al*, etc.
- Large scale generation takes months running at  $O(10)$  Tflops
  - Capability computing
  - Current computations use BG / Cray etc.
- Can GPU clusters make headway in this regime?



(Clusters dedicated to lattice QCD at Fermilab and Jefferson Lab)

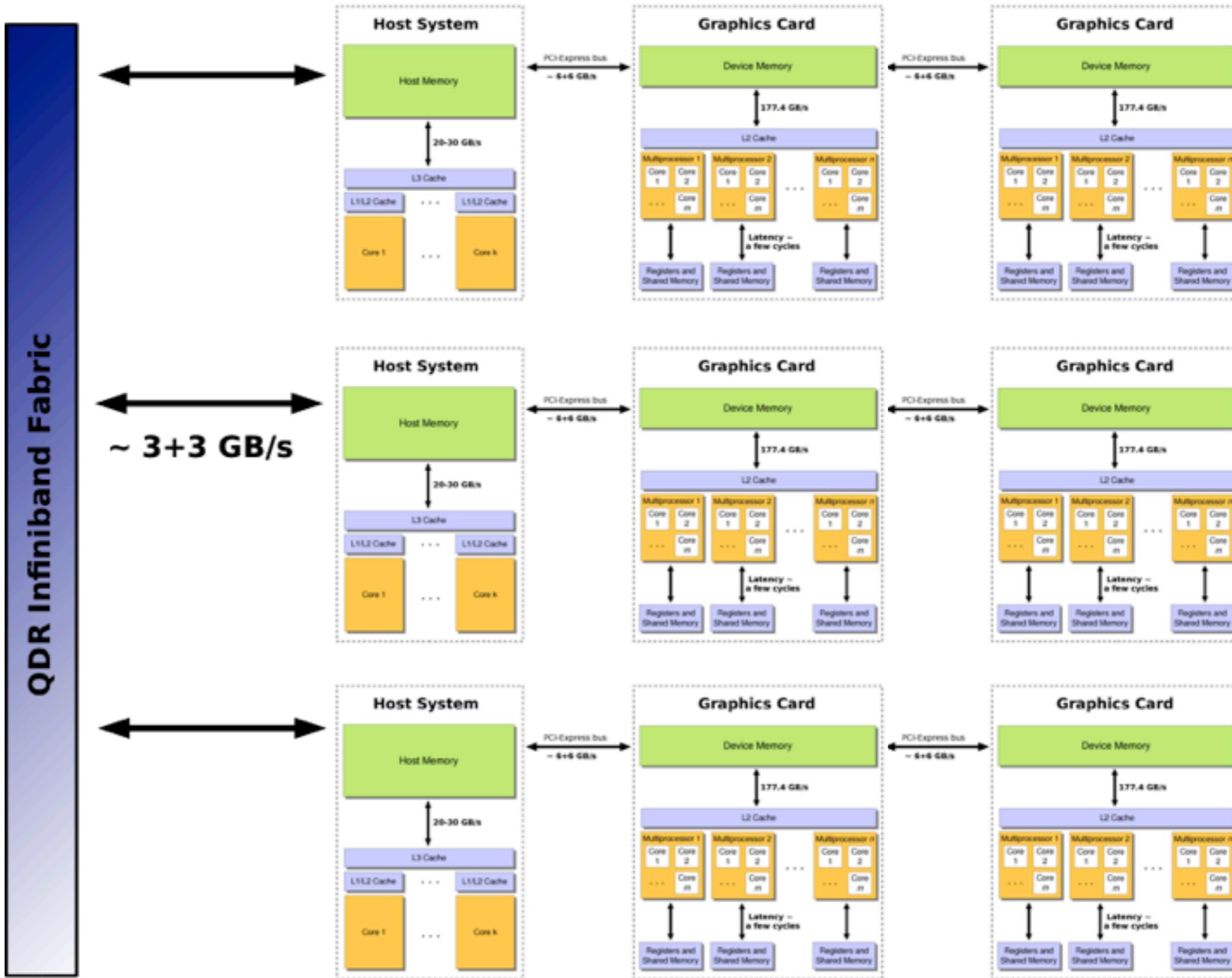


"Intrepid" - Argonne Leadership Computing Facility



"Jaguar" - Oak Ridge Leadership Computing Facility

# Memory Hierarchy



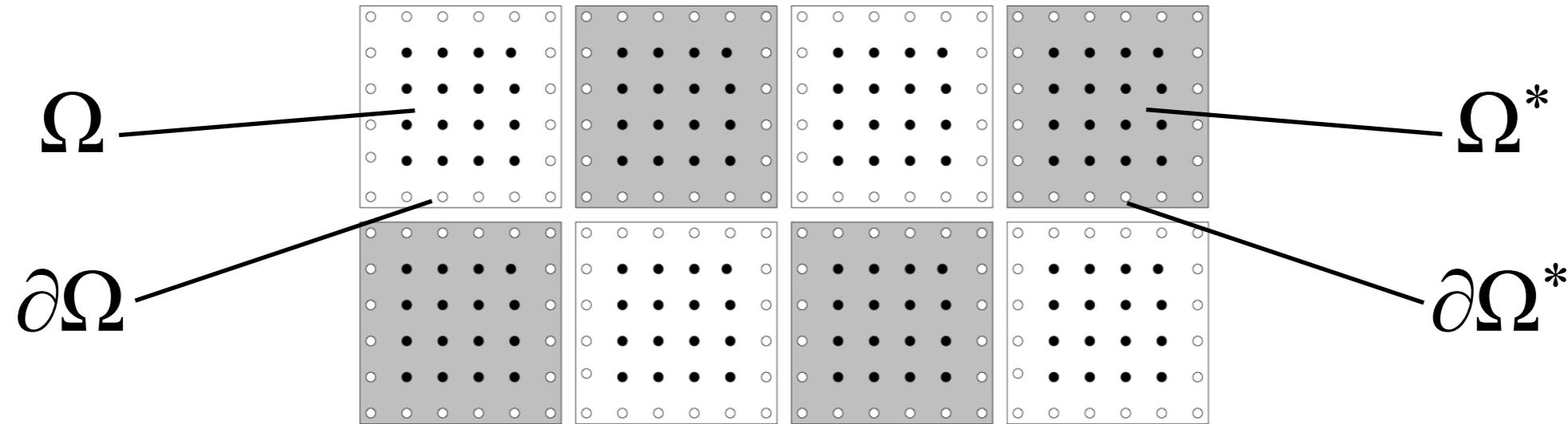
# Domain Decomposition

---

- Only way to get to O(1000) GPUs is with communication reducing algorithms
- Schwarz Methods aka Domain Decomposition
  - Partition the lattice into domains
  - Solve each of the domains independently
  - Use domain solutions to correct full system
- Typically used a preconditioner
- Previous work in LQCD
  - Lüscher: SAP (multiplicative Schwarz) on CPU clusters
  - Osaki and Ishikawa: additive Schwarz for GPU scaling up to 8 GPUs
  - Nakamura *et al*: SAP on QPACE, ~12 Tflops at 256 Cells

# Multiplicative Schwarz

---

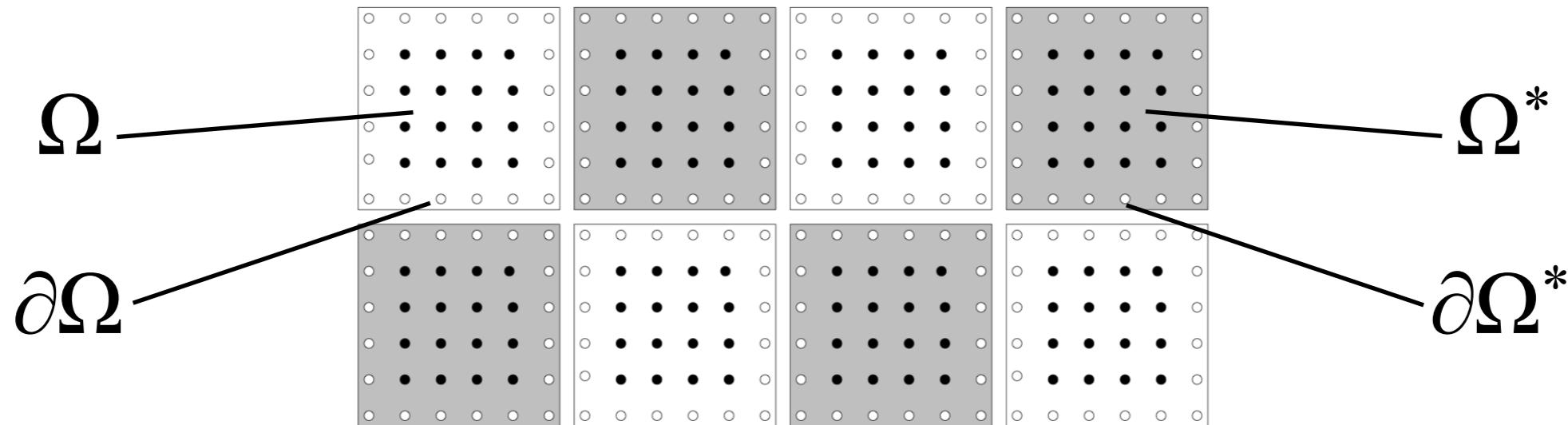


- Update domains sequentially and use most recent solutions **Block Gauss Seidel**
- Coloring algorithms used to parallelize, **but this means either**
  - Multiple colors per processor
  - Idling processors
- Preconditioner given by

$$K = D_{\Omega}^{-1} + D_{\Omega^*}^{-1} + D_{\Omega^*}^{-1} D_{\partial\Omega^*} D_{\Omega}^{-1}$$

# Additive Schwarz

---

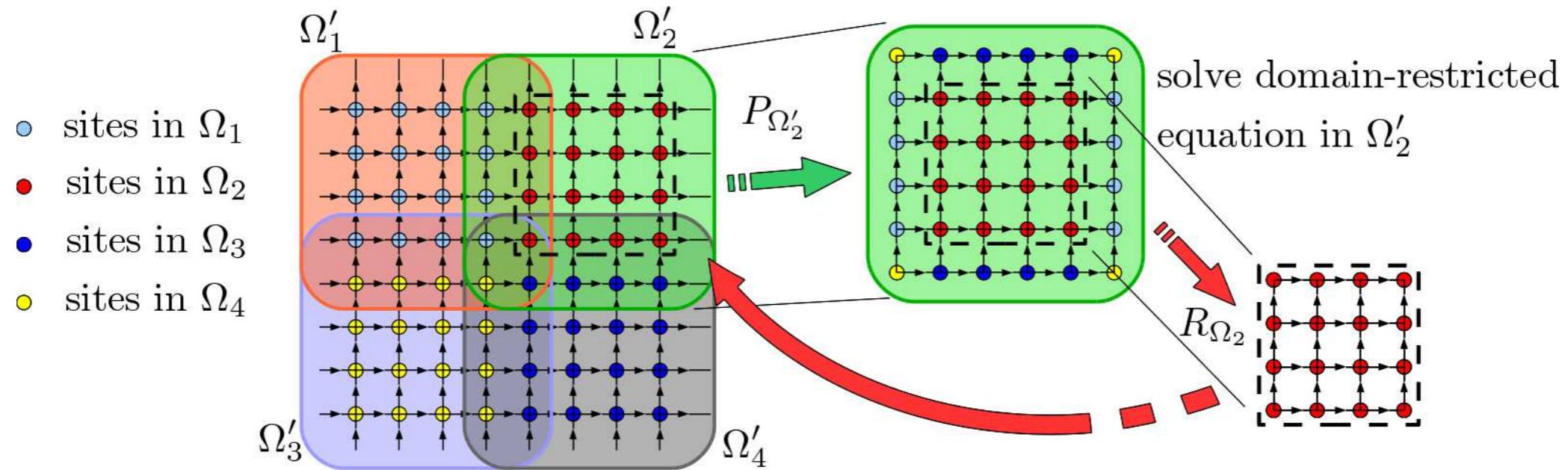


- Update domains simultaneously **Block Jacobi**
- Expect higher iteration count than multiplicative
- Algorithm is trivially parallel
  - Better suited to GPU architecture
- Preconditioner given by

$$K = D_{\Omega}^{-1} + D_{\Omega^*}^{-1}$$

# Restricted Additive Schwarz

(Cai and Sarkis)



- Initial domain residuals given by the restriction of full system residual
- Impose Dirichlet boundary conditions at domain boundaries
- Solve all domain systems simultaneously
- Full system correction given by the sum of the domain solutions

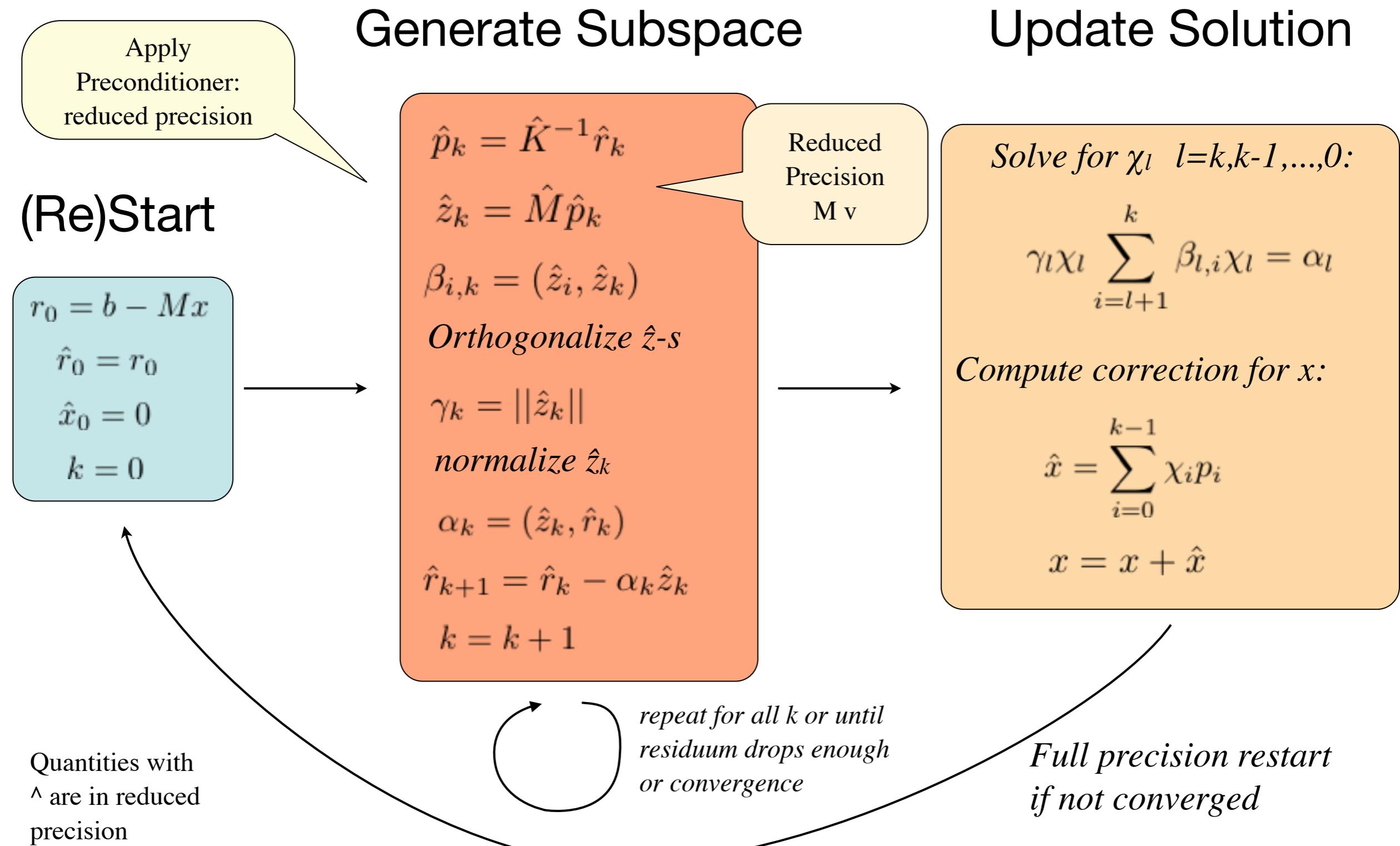
figure taken from Osaki and Ishikawa

# Implementing the Preconditioner

---

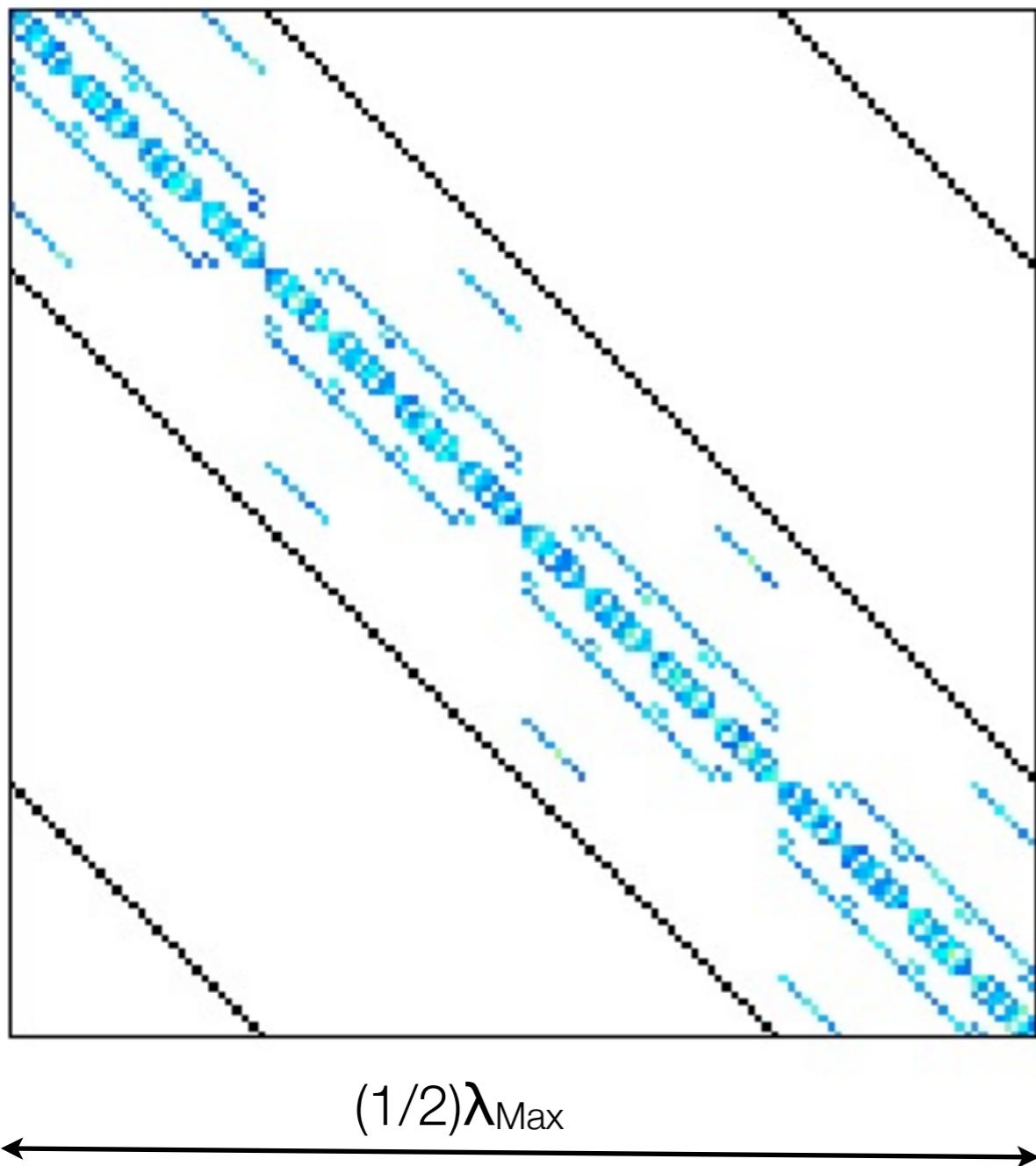
- Non-overlapping domains only
- Domain volume = local GPU volume
  - Solver iteration count varies with  $N_{\text{GPU}}$
- Very easy to implement
  - Switch off inter-GPU communication for restricted Dirac operator
- 10 steps of Minimum residual (MR) used to solve for preconditioner
- Preconditioner is a gross approximation
  - Only need 16-bit precision

# Mixed Precision GCR-DD solver



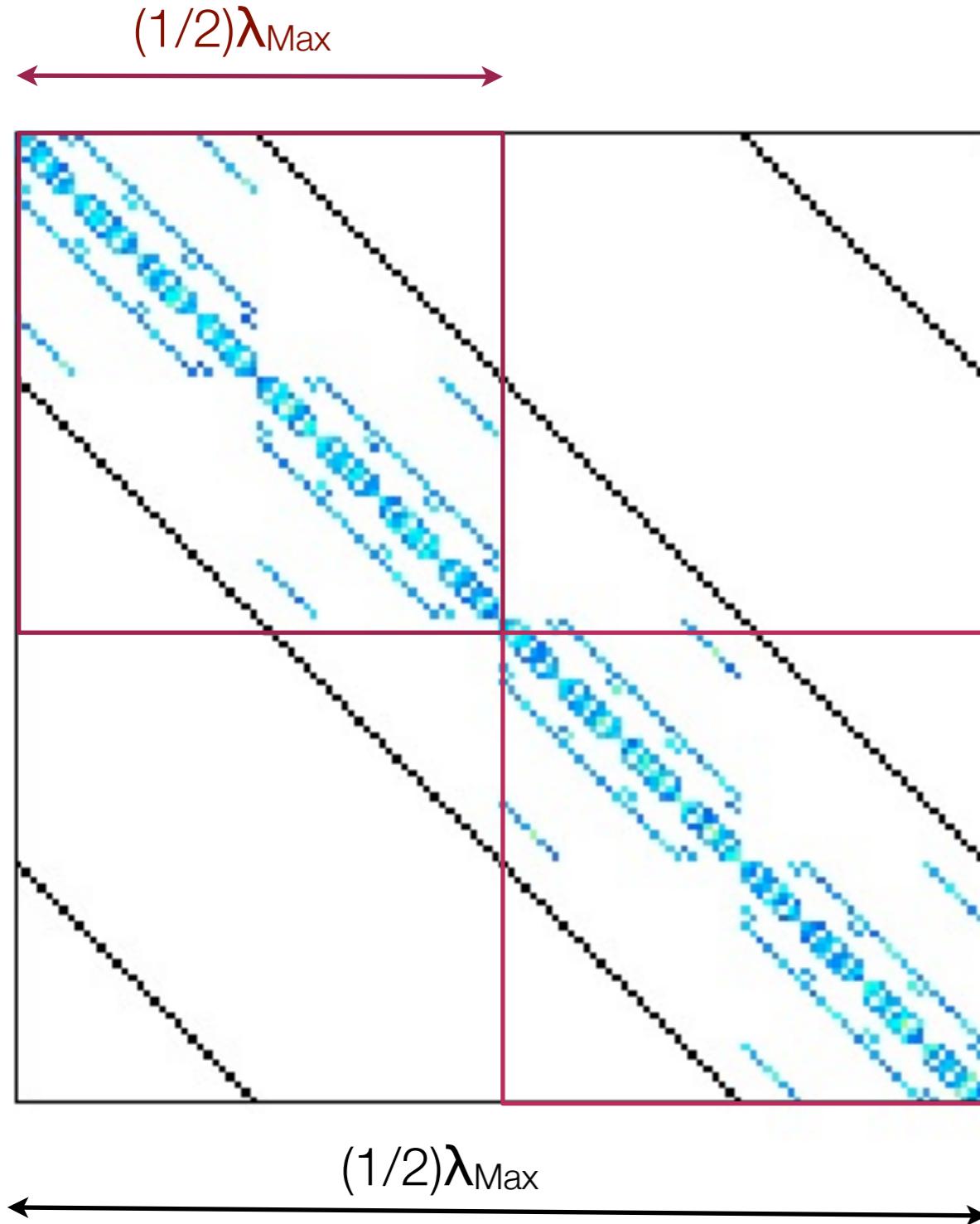
# Size Matters

---



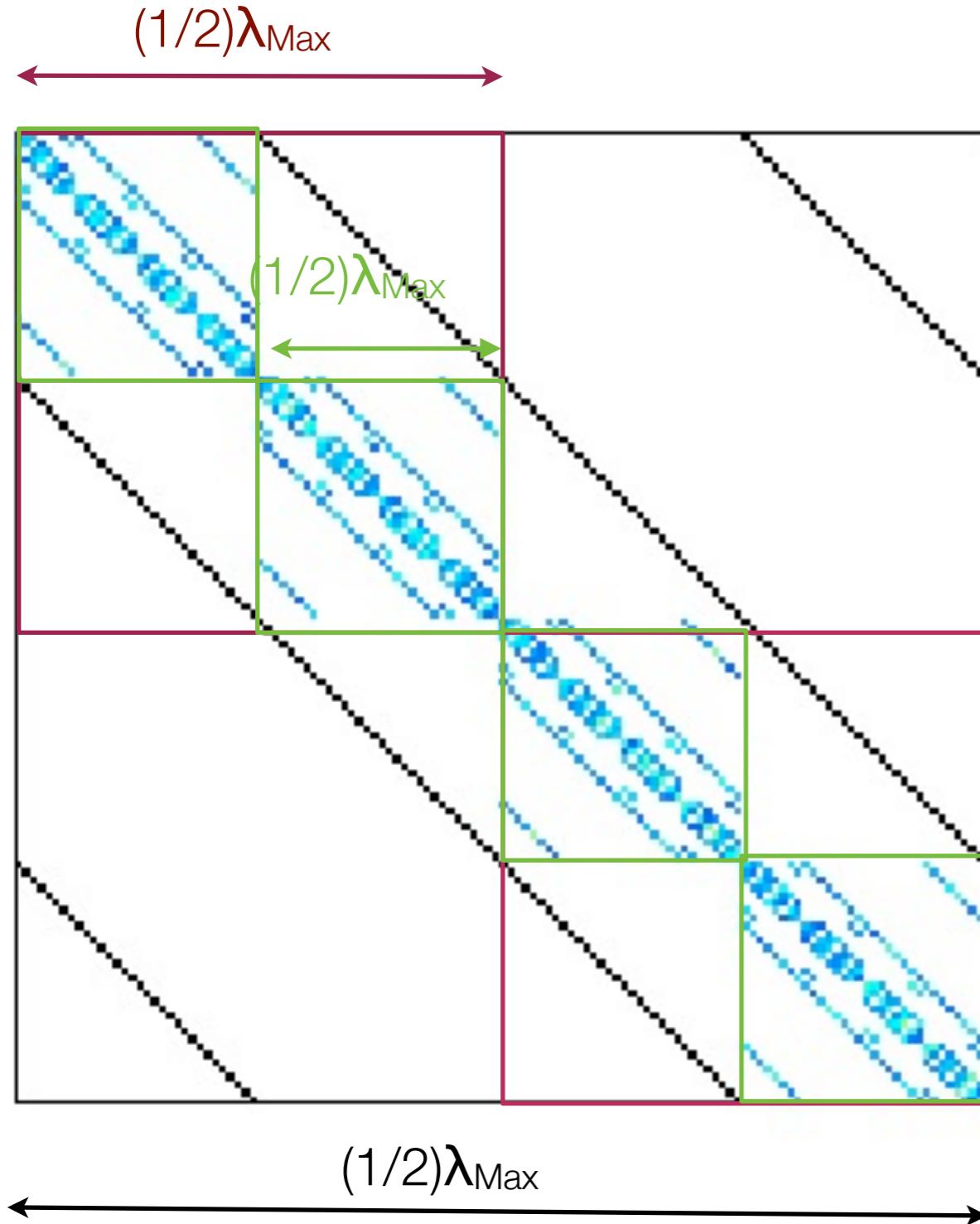
- No comms between domains
  - Block Diagonal Preconditioner
- Blocks impose  $\lambda$  cutoff
- Finer Blocks
  - lose structure in operator
  - lose long wavelength/low energy modes
- Heuristically (& from Lüscher)
  - keep wavelengths of  $\sim O(\Lambda_{\text{QCD}}^{-1})$
  - $\Lambda_{\text{QCD}}^{-1} \sim 1\text{fm}$
  - $32^3 \times 256$  Aniso: ( $a_s=0.125\text{fm}$ ,  $a_t=0.035\text{fm}$ )
    - Our case:  $8^3 \times 32$  blocks are ideal
  - Iso:  $1\text{fm} \sim 8\text{-}10$  sites ( $a=0.11\text{fm}$ )
- Min. blocksize has scaling implications

# Size Matters



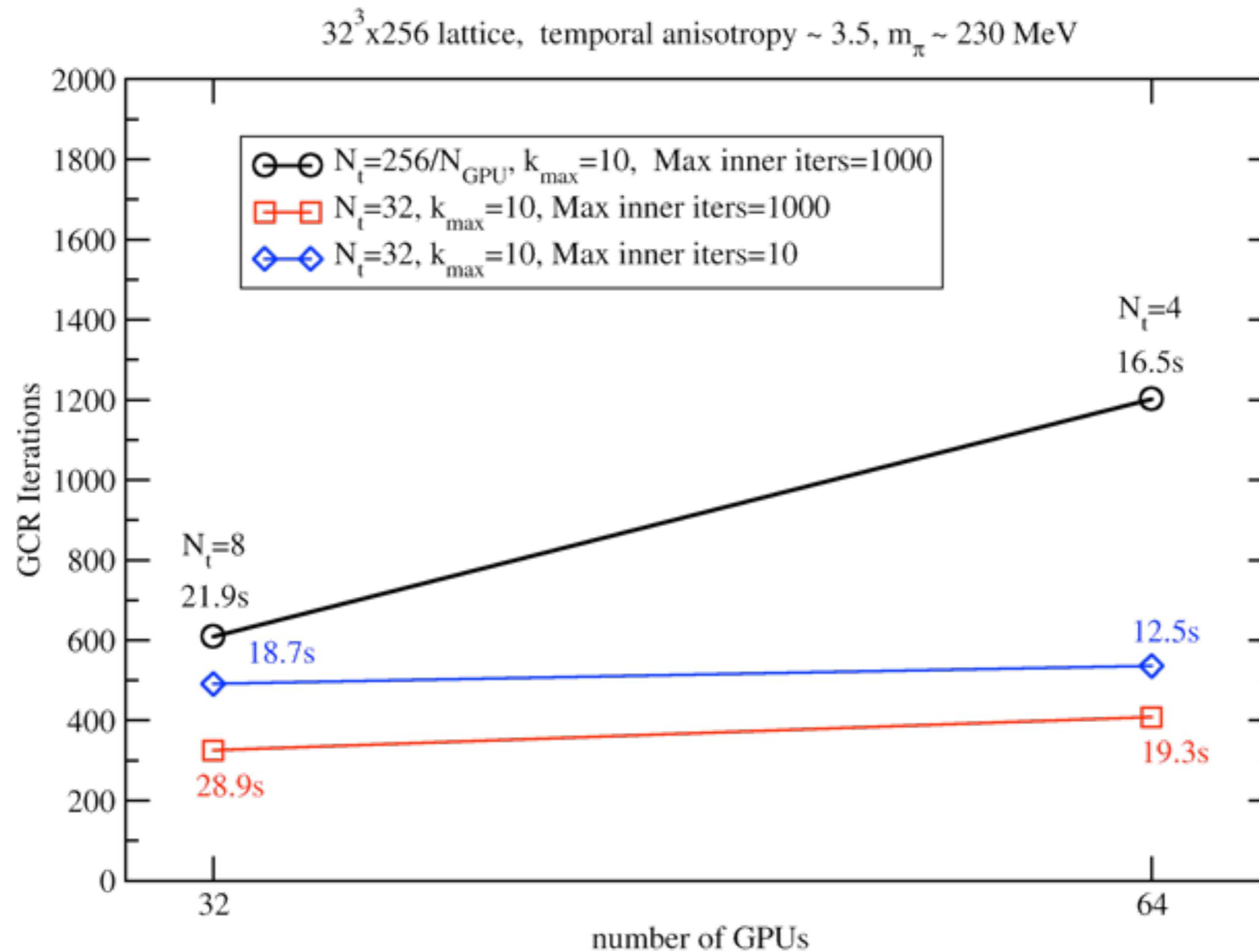
- No comms between domains
  - Block Diagonal Preconditioner
  - Blocks impose  $\lambda$  cutoff
  - Finer Blocks
    - lose structure in operator
    - lose long wavelength/low energy modes
- Heuristically (& from Lüscher)
  - keep wavelengths of  $\sim O(\Lambda_{\text{QCD}}^{-1})$
  - $\Lambda_{\text{QCD}}^{-1} \sim 1\text{fm}$
  - $32^3 \times 256$  Aniso: ( $a_s=0.125\text{fm}$ ,  $a_t=0.035\text{fm}$ )
    - Our case:  $8^3 \times 32$  blocks are ideal
  - Iso:  $1\text{fm} \sim 8\text{-}10$  sites ( $a=0.11\text{fm}$ )
- Min. blocksize has scaling implications

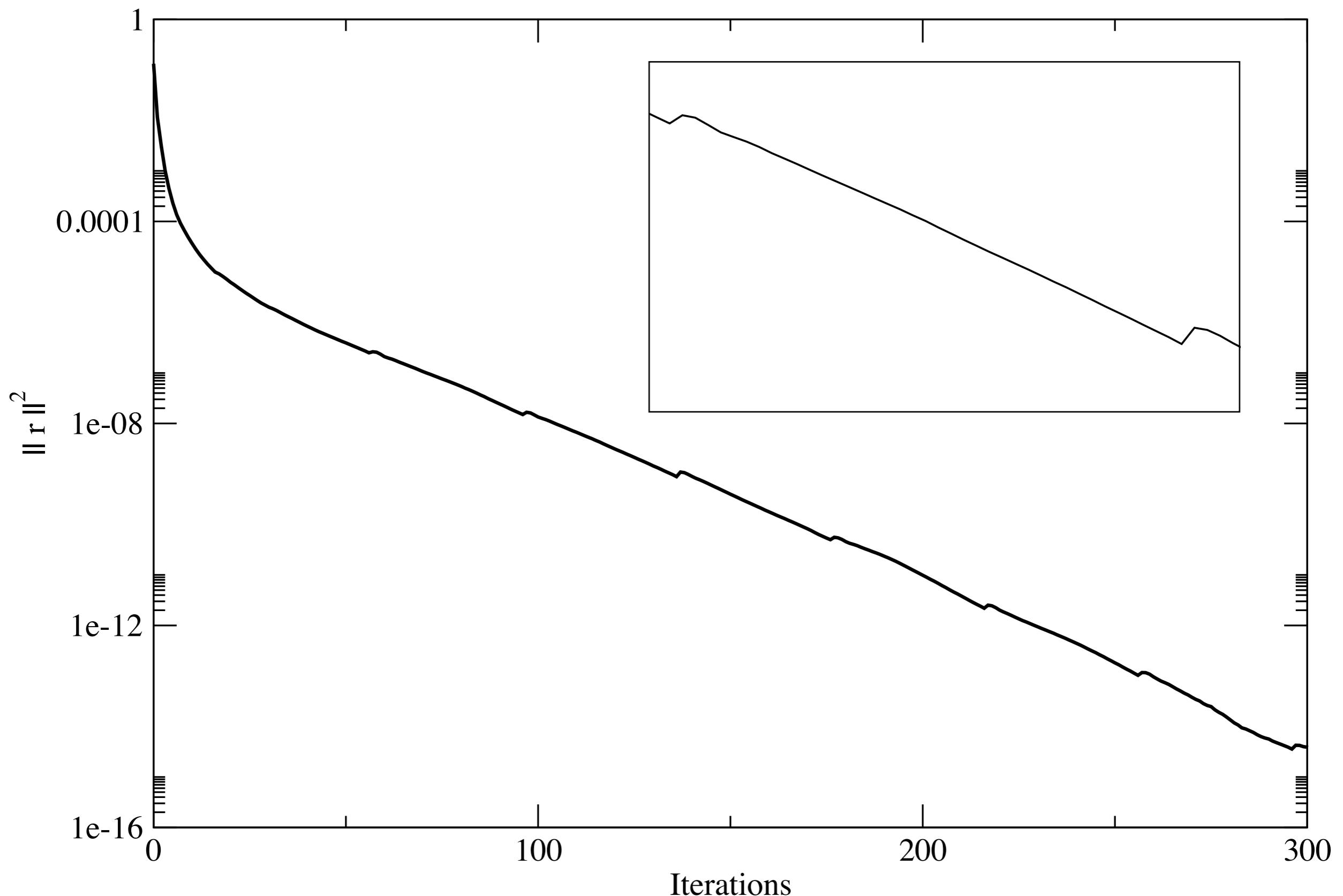
# Size Matters

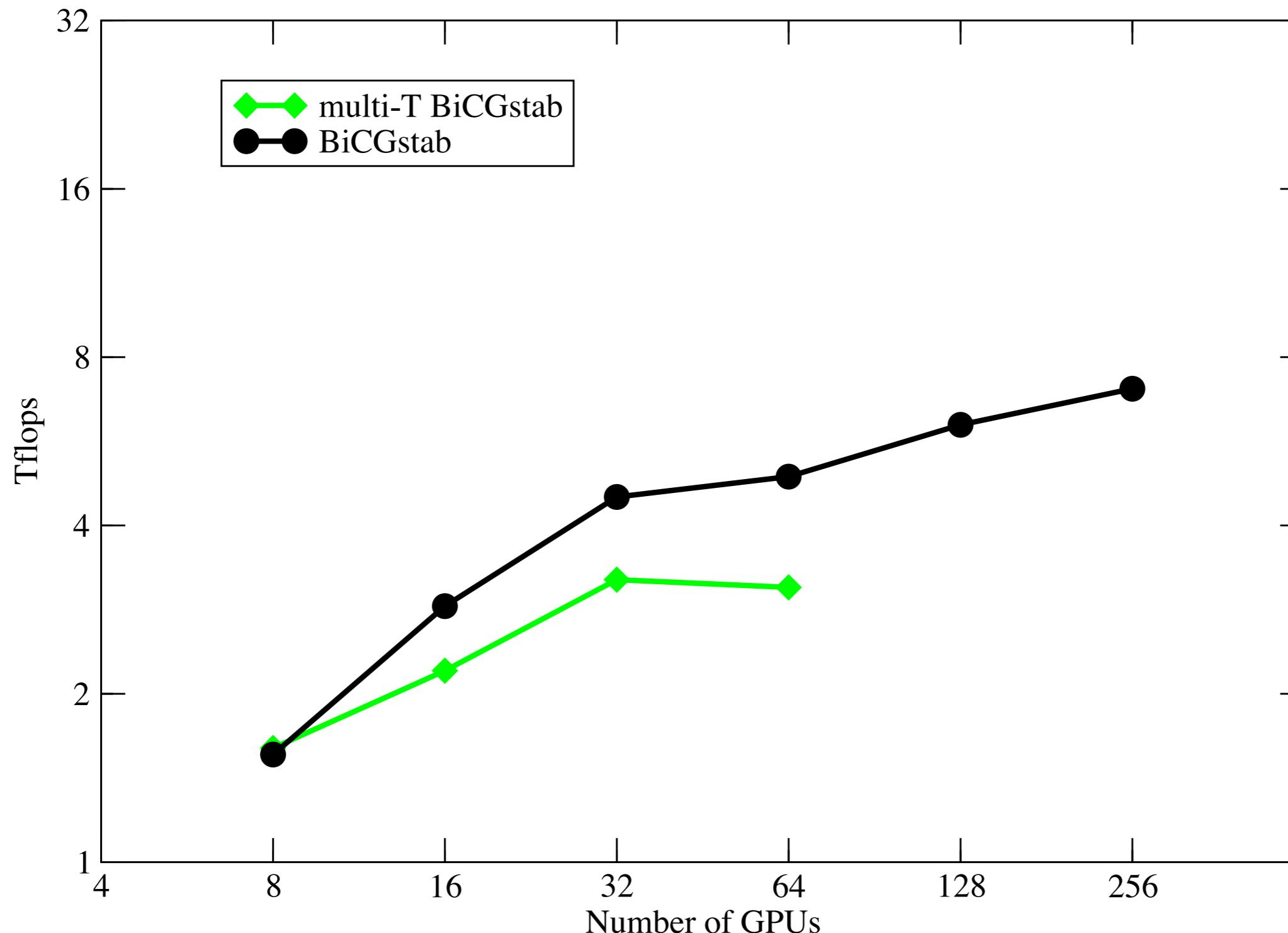


- No comms between domains
  - Block Diagonal Preconditioner
  - Blocks impose  $\lambda$  cutoff
  - Finer Blocks
    - lose structure in operator
    - lose long wavelength/low energy modes
- Heuristically (& from Lüscher)
  - keep wavelengths of  $\sim O(\Lambda_{\text{QCD}}^{-1})$
  - $\Lambda_{\text{QCD}}^{-1} \sim 1\text{fm}$
  - $32^3 \times 256$  Aniso: ( $a_s=0.125\text{fm}$ ,  $a_t=0.035\text{fm}$ )
    - Our case:  $8^3 \times 32$  blocks are ideal
  - Iso:  $1\text{fm} \sim 8\text{-}10$  sites ( $a=0.11\text{fm}$ )
- Min. blocksize has scaling implications

# Size Matters

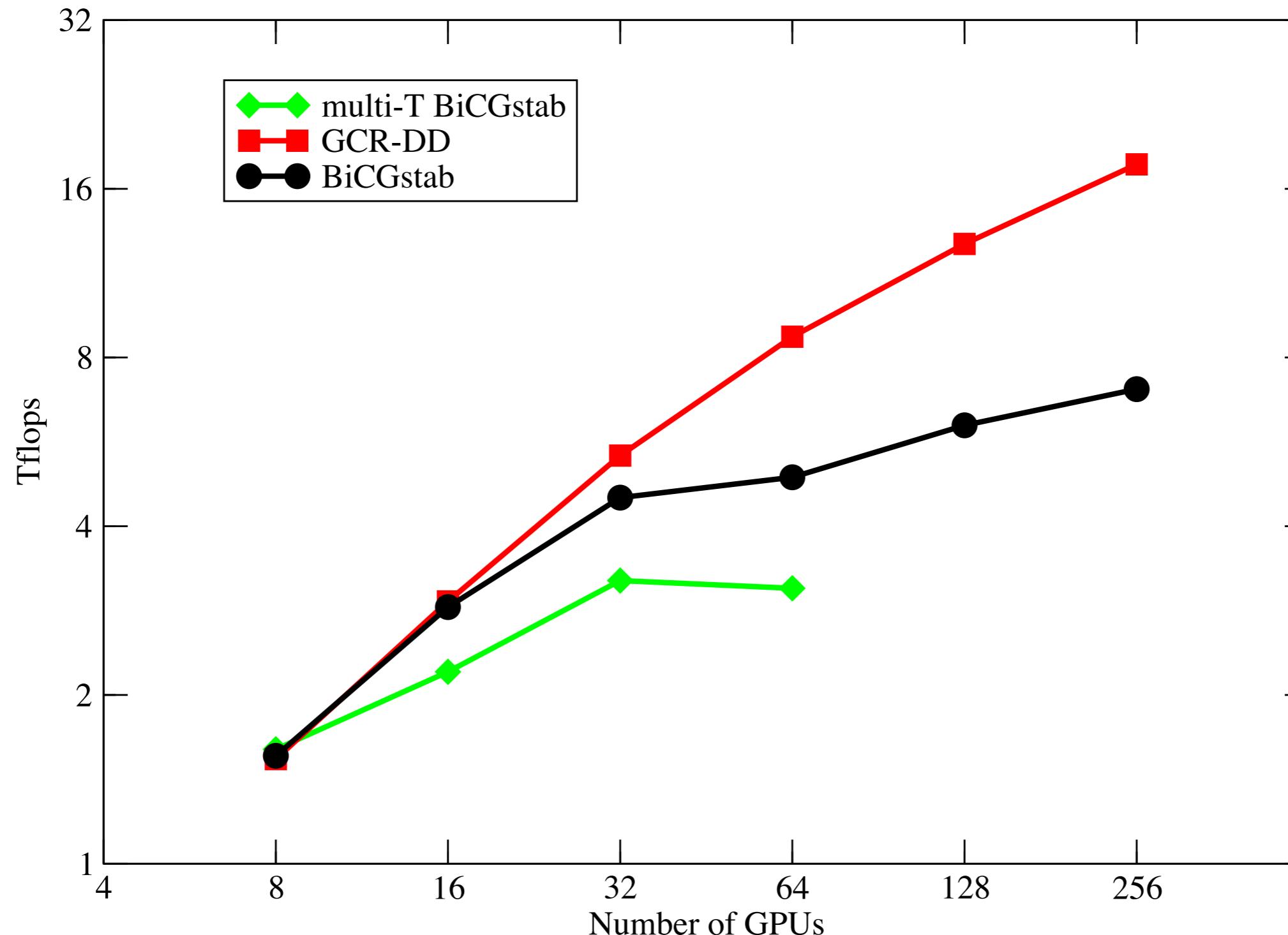






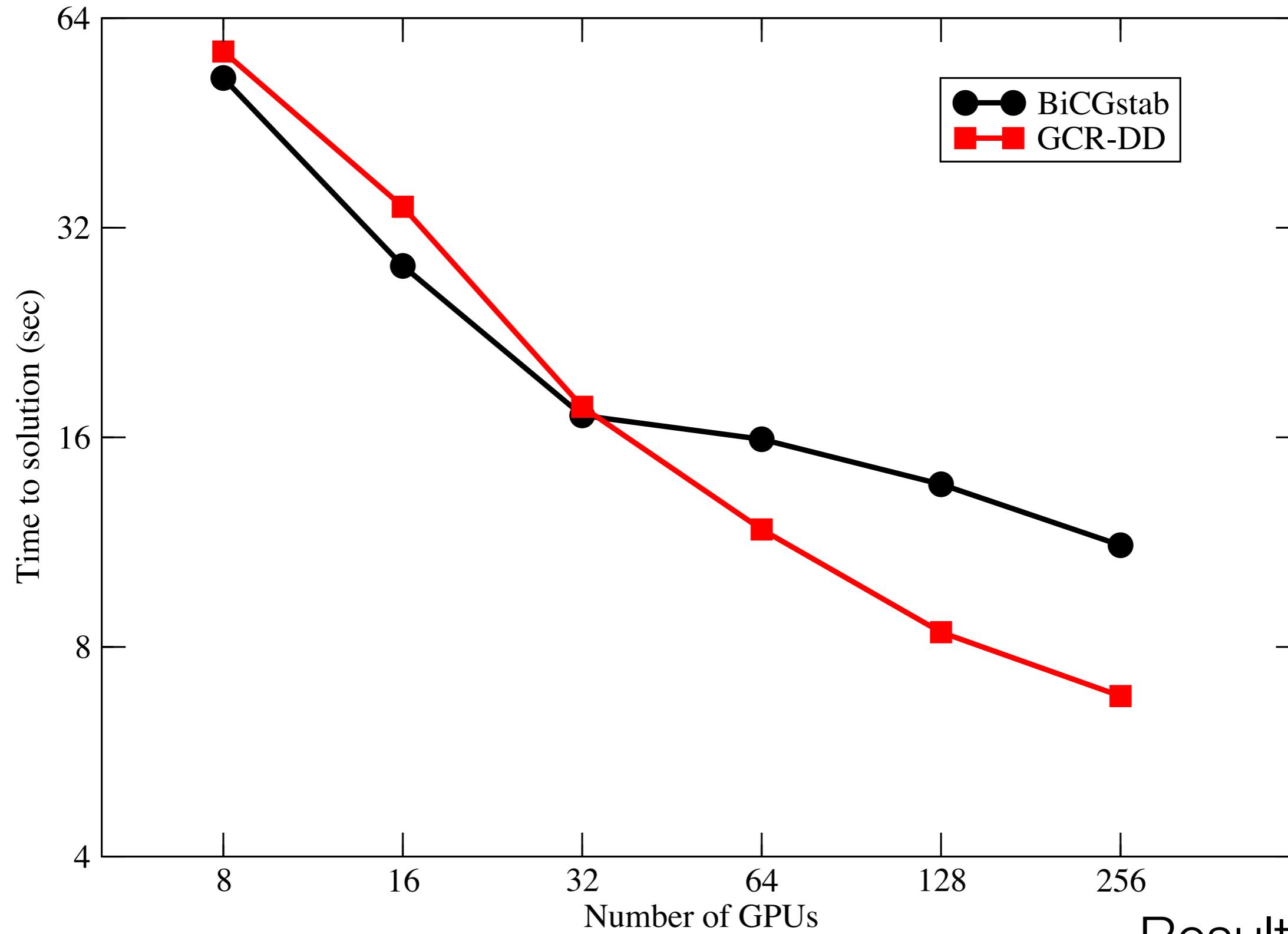
Wilson clover  
32<sup>3</sup>x256

Results on  
Edge @ LLNL

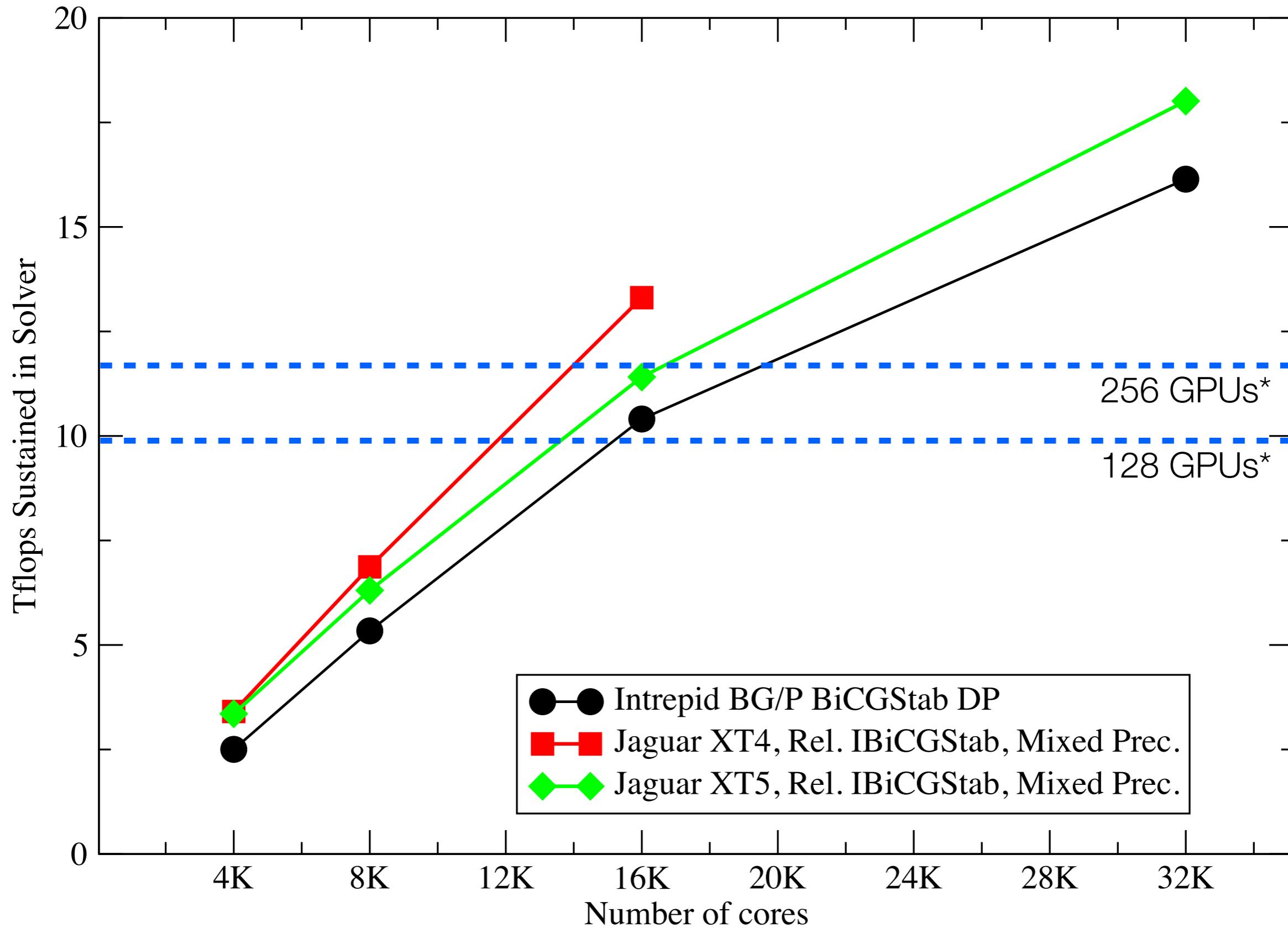


Wilson clover  
32<sup>3</sup>x256

Results on  
Edge @ LLNL



Results on  
Edge @ LLNL

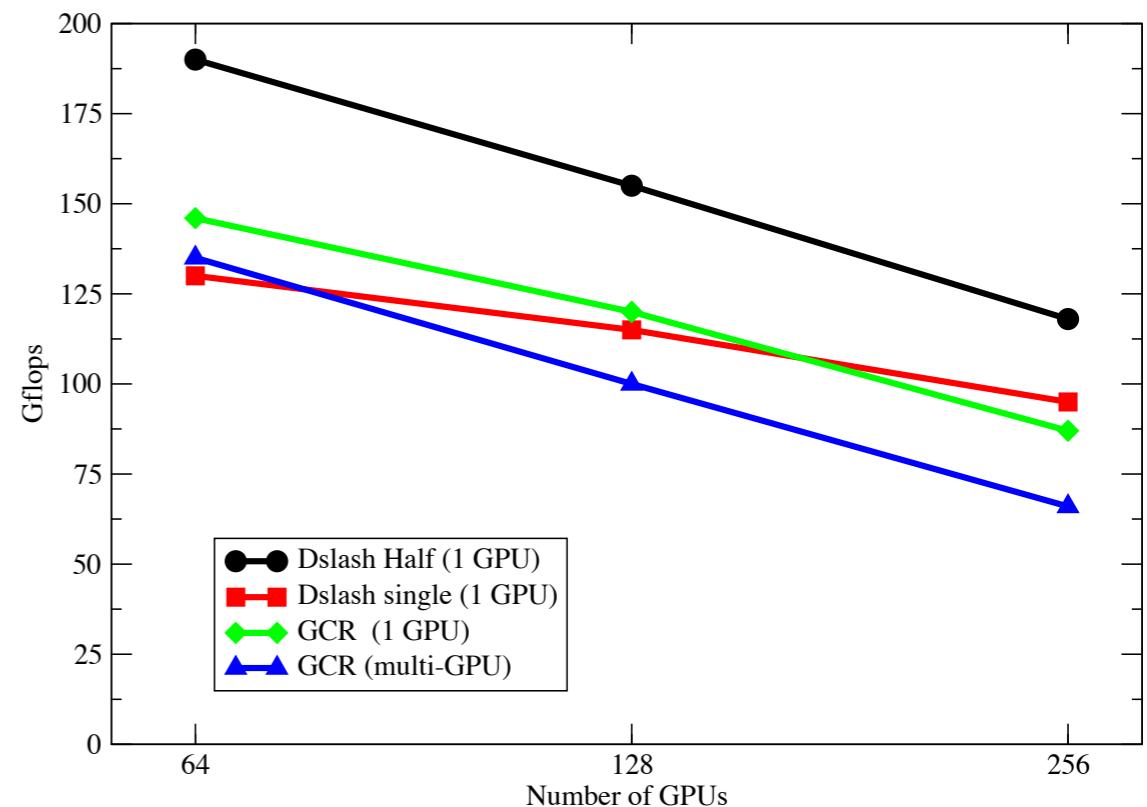


\*GPU Tflops scaled according solver iterations

# Please sir, can I have some more?

---

- 17.2 Tflop on 256 GPUs = 69 Gflops/GPU
  - using all the precision tricks
  - local problem is small
    - Low occupancy?
    - Driver overheads?
    - Communications?
- $40^3 \times 256$ , 2x bigger V
  - Expect 21 Tflops if comms bound
  - Obtain 24 Tflops
- Performance bound by small local volume
  - Solve multiple systems simultaneously
  - Optimize for small volumes?

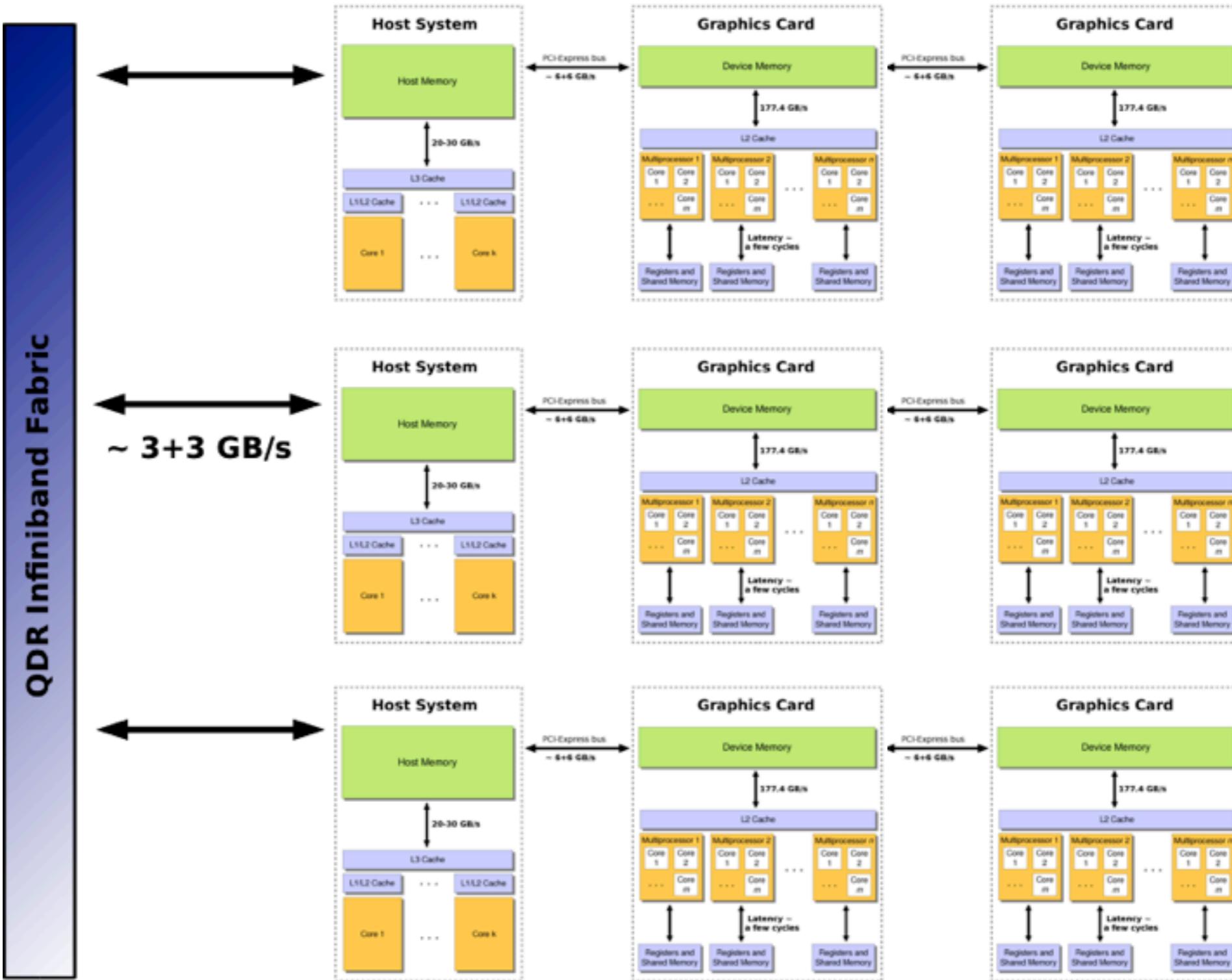


# Refining the Algorithm

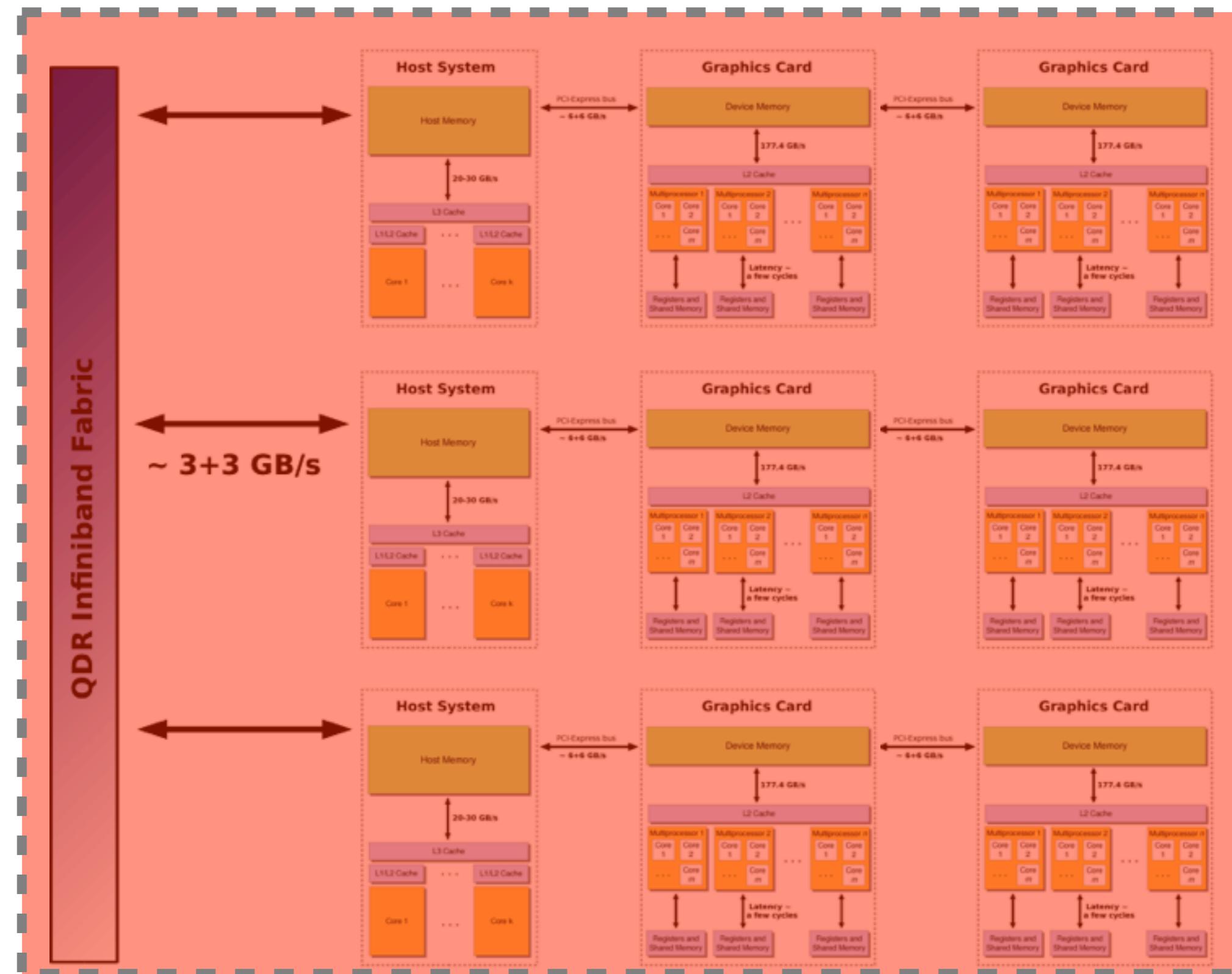
---

- RAS preconditioner = block Jacobi
  - Expect acceleration from over-relaxed preconditioner
- Preconditioner only requires  $\tau > 0.1$ 
  - Use 8-bit precision?
- No incorporation of GPU Direct yet
  - Modest improvement expected since not totally comms bound
- Multi-level domain decomposition

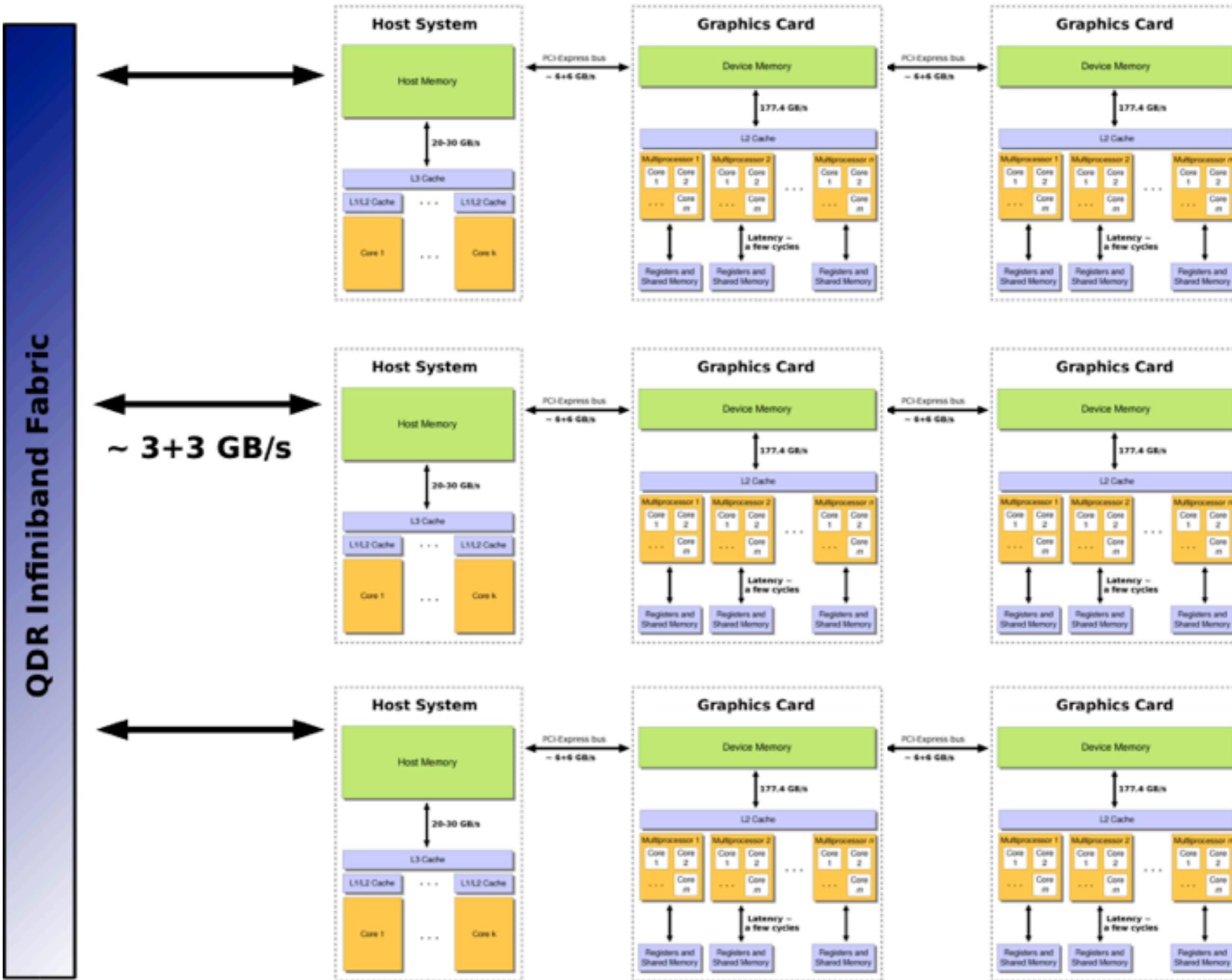
# Multi-level domain decomposition



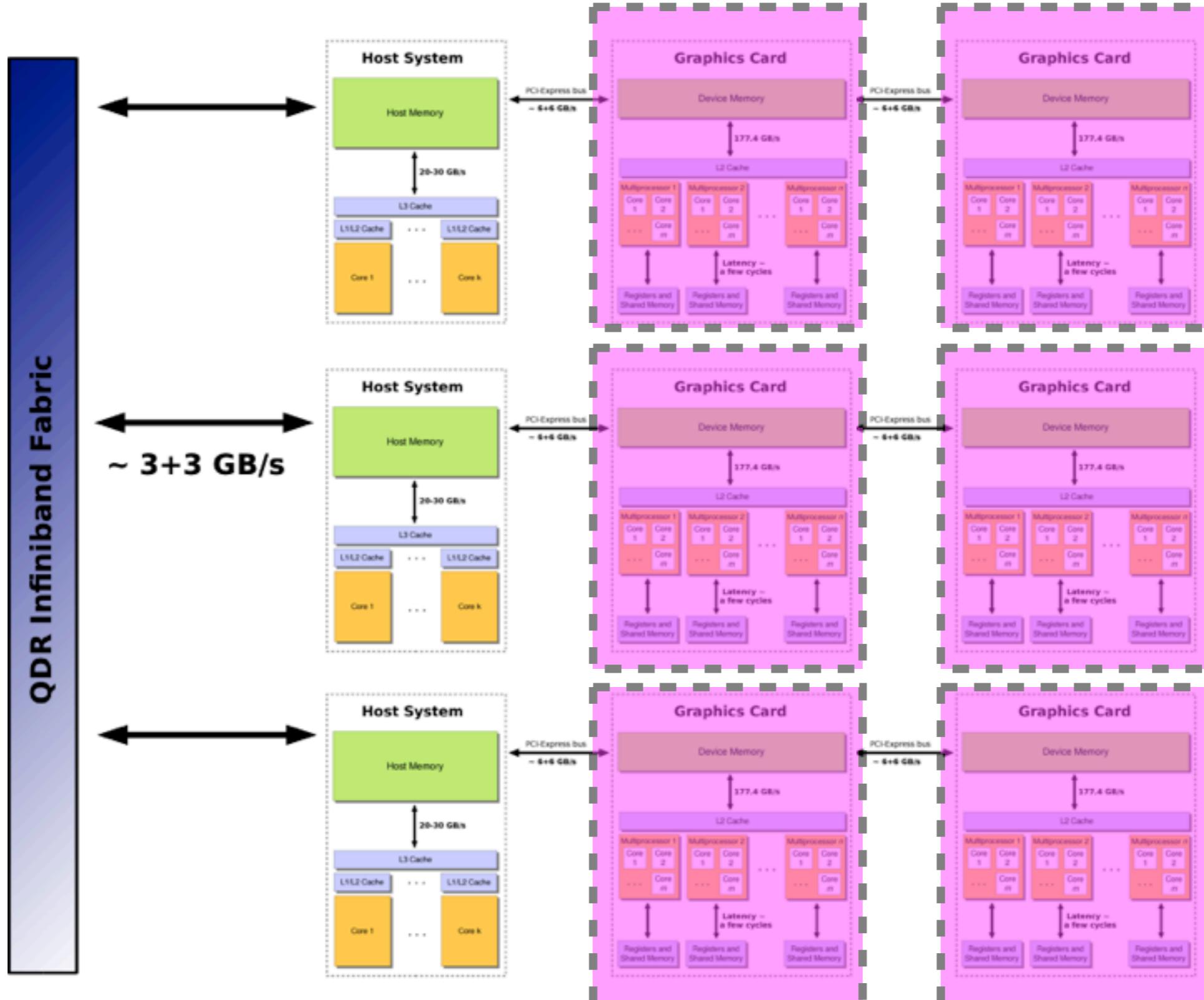
# Multi-level domain decomposition



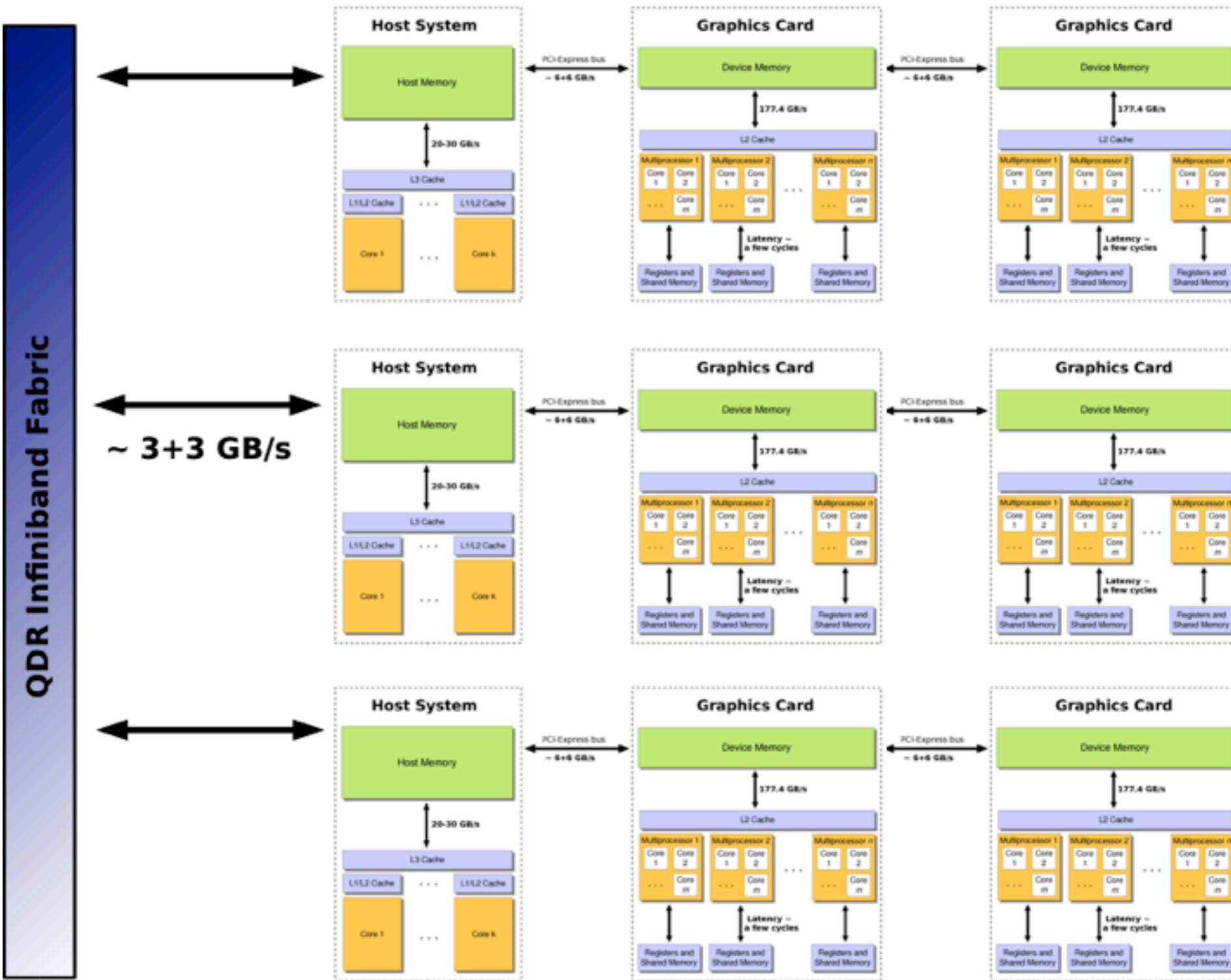
# Multi-level domain decomposition



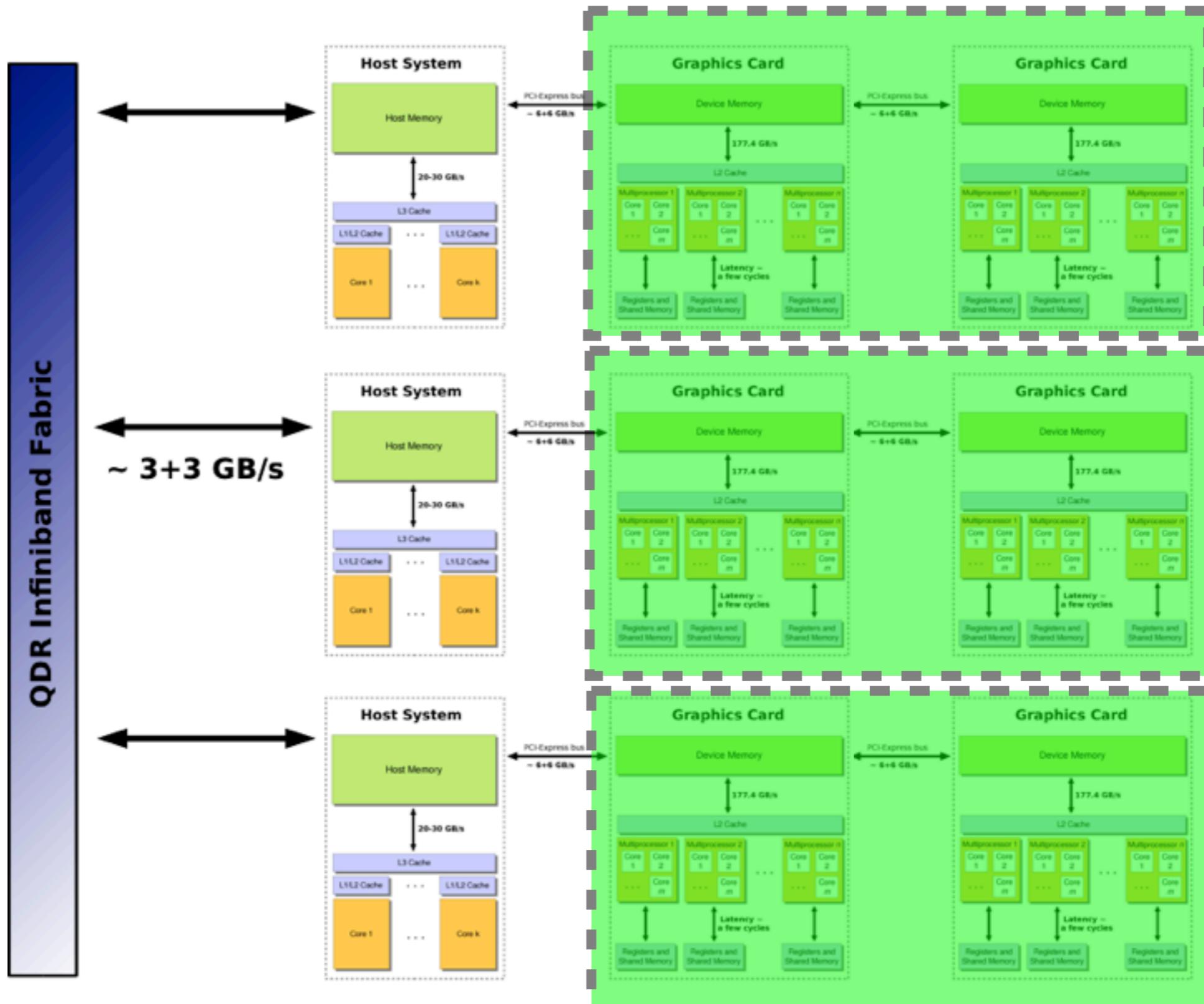
# Multi-level domain decomposition



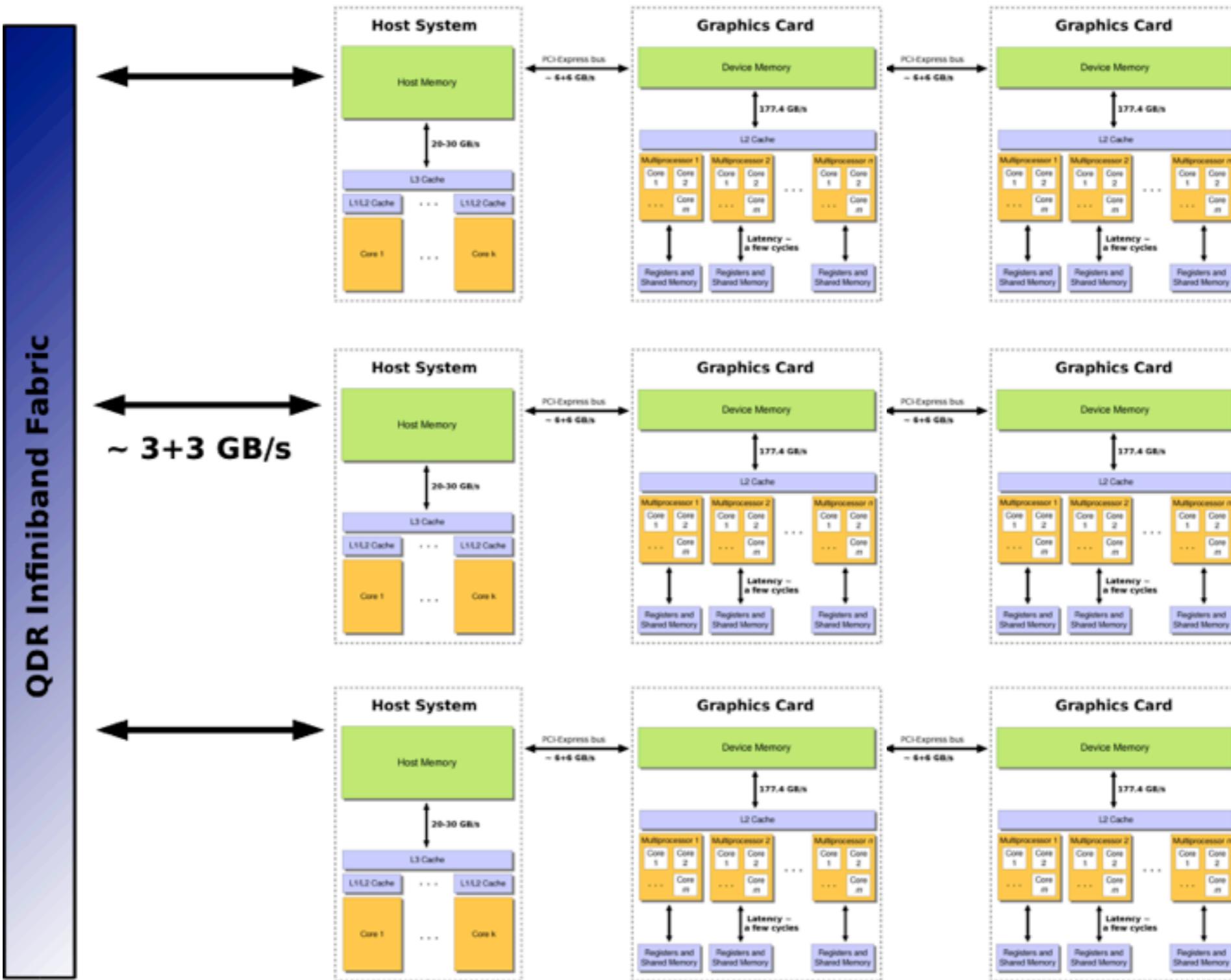
# Multi-level domain decomposition



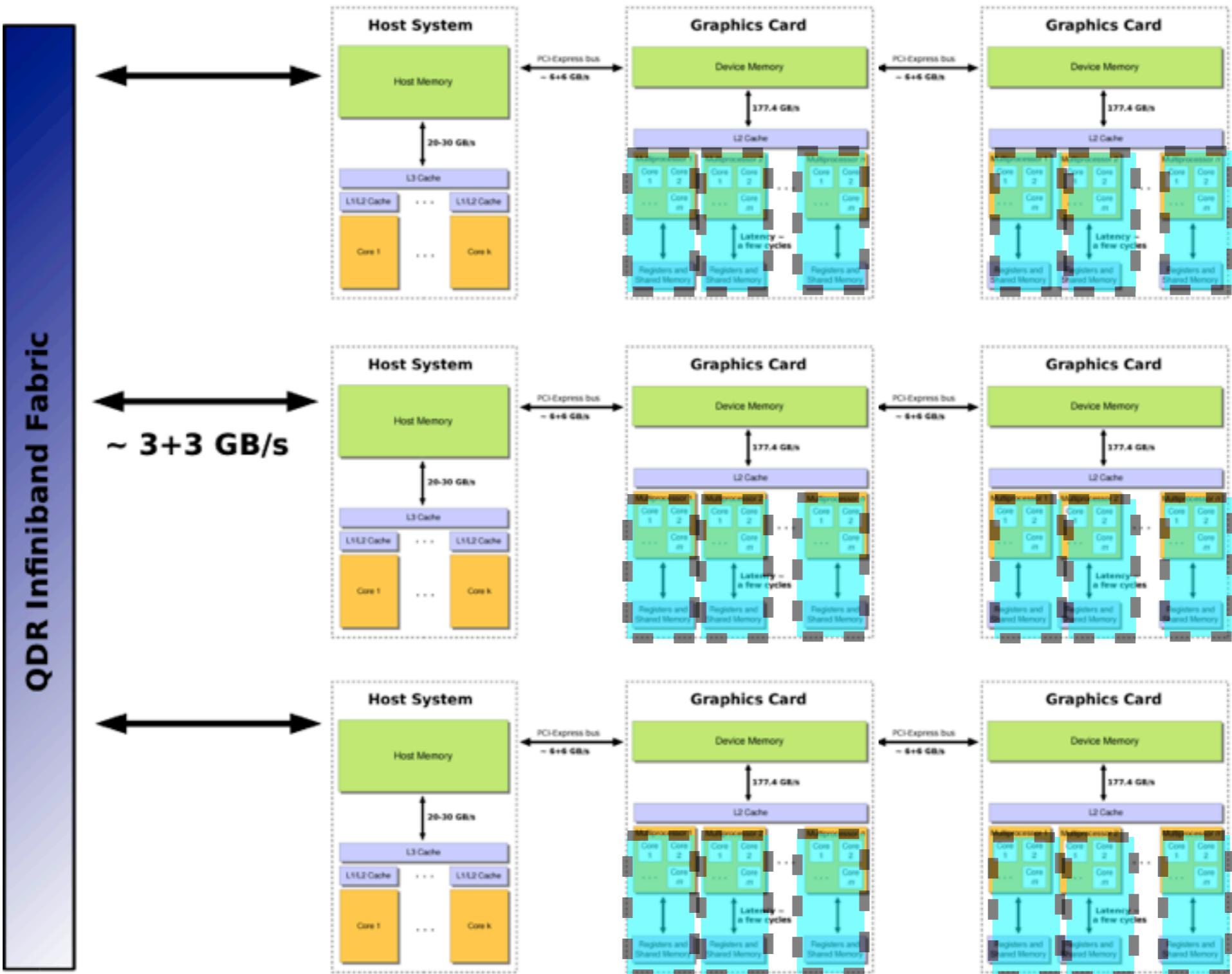
# Multi-level domain decomposition



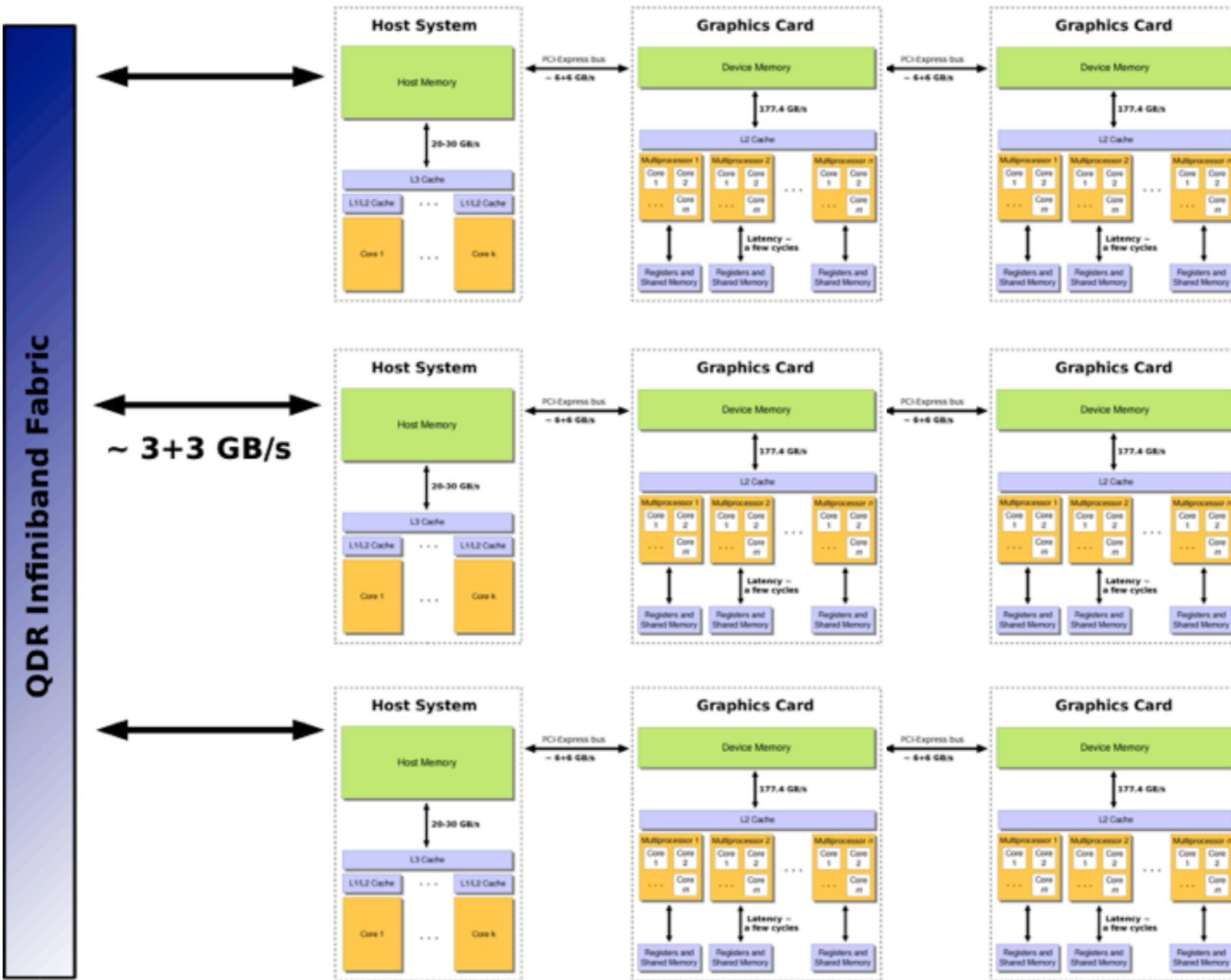
# Multi-level domain decomposition



# Multi-level domain decomposition



# Multi-level domain decomposition



# Conclusions and Outlook

---

- First demonstration of QCD GPU computation at the 10+ Tflops scale
  - To appear in **Proceedings of SC11**, “Scaling lattice QCD past 100 GPUs”
- Communication reduction algorithms critical for strong scaling
  - Flops are cheaper than bytes
- GPU clusters **can** replace capability super computers
- Huge algorithmic parameter space to explore
  - Overlapping blocks, boundary conditions, multi-level
- DD preconditioner  $\Rightarrow$  smoother for  $\alpha$ MG

# <https://github.com/lattice/quda>

The screenshot shows the GitHub Issues page for the `lattice/quda` repository. The user is logged in as `mikeaclarck` with 29 notifications. The repository name is `lattice / quda`. The Issues tab is selected, showing 3 open issues. The Issues list includes:

- #27 Objectify the solvers feature by mikeaclarck April 27, 2011
- #26 Objectify and generalize the gauge field clean-up feature by mikeaclarck April 27, 2011
- #24 Add support for QIO feature by mikeaclarck April 20, 2011

Filters available include `Submitted`, `Updated`, and `Comments`. A search bar at the top right allows for searching issues and milestones.

Everyone's Issues	3
Assigned to you	3
Mentioning you	0
Milestone: Adaptive Multigrid	<span style="font-size: small;">⚙️</span>
3 open issues	
<hr/>	
LABELS	
<span style="color: purple;">█</span> clean-up	1
<span style="color: green;">█</span> feature	3

# <https://github.com/lattice/quda>

Screenshot of the GitHub network graph for the lattice/quda repository.

The screenshot shows the GitHub interface for the repository `lattice/quda`. The user is logged in as `mikeaclarke` (29 notifications). The main navigation bar includes links for `Dashboard`, `Inbox`, `Account Settings`, and `Log Out`. Below the navigation is a search bar and links for `Explore GitHub`, `Gist`, `Blog`, and `Help`.

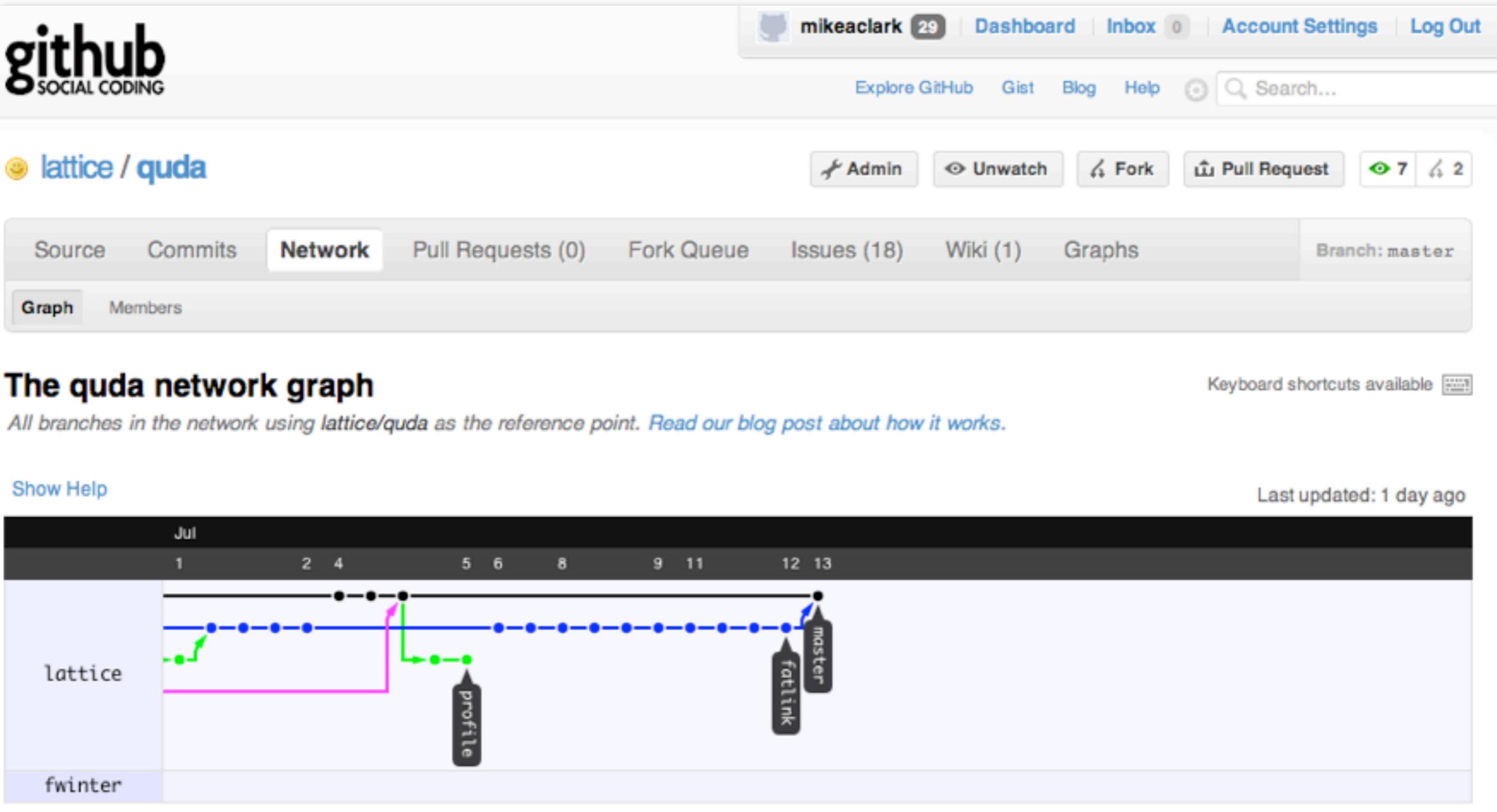
The repository name `lattice / quda` is displayed, along with buttons for `Admin`, `Unwatch`, `Fork`, `Pull Request`, and metrics showing 7 issues and 2 pull requests.

The top navigation bar has tabs for `Source`, `Commits`, `Network` (selected), `Pull Requests (0)`, `Fork Queue`, `Issues (18)`, `Wiki (1)`, and `Graphs`. The branch is set to `master`.

The main content area is titled **The quda network graph**. It displays a timeline from July 1st to July 13th, showing the evolution of branches. A pink box highlights a specific commit on the `lattice` branch at approximately July 4th. A green arrow points to a commit on the `profile` branch at approximately July 5th. A blue dashed line connects the `lattice` and `profile` branches. A red arrow points to a commit on the `fatlink` branch at approximately July 12th. A black arrow points to a commit on the `master` branch at approximately July 13th.

Below the graph, there is a note: *All branches in the network using lattice/quda as the reference point. Read our blog post about how it works.*

Buttons for `Show Help` and `Last updated: 1 day ago` are visible.



# <https://github.com/lattice/quda>

Screenshot of the GitHub repository for lattice/quda, showing the commit history from July 11 to July 13, 2011.

The repository owner is mikeaclarke (29 notifications). The repository name is lattice/quda. The current branch is master. There are 18 issues and 1 wiki page.

The commit history shows the following changes:

- 2011-07-13**
  - Merge branch 'fatlink'  
Guochun Shi (author) 1 day ago
  - commit 1dea1faeb6c8f04ca6f39  
tree deaa1f0bb4e0cae2e9e5  
parent 2d033b37514e5ffa6ca9  
parent 65f3a5c67c9d2ff52bf2
- 2011-07-12**
  - put exterior and gather kernel together and reverse orders for exterior ...  
Guochun Shi (author) 1 day ago
  - commit 65f3a5c67c9d2ff52bf2  
tree 85cc5f0278919ebbc161  
parent 8fcba92cd2f65911e5f09
- 2011-07-11**
  - print out an error message about not supporting odd dim size and 12-reco...  
Guochun Shi (author) 2 days ago
  - commit 8fcba92cd2f65911e5f09  
tree 8f29ac819e670b0ea8ef  
parent dbacb526c5f515a277d5
  - code in anti-periodic phase for link flattening as found in MILC  
Guochun Shi (author) 2 days ago
  - commit dbacb526c5f515a277d5  
tree 632cd3a42b97b5737aec  
parent 971ea91a1ca0fb2f708e

<https://github.com/lattice/quda>

Come join the fun...