29th International Conference on Parallel Computational Fluid Dynamics
May 15-17, 2017; Glasgow, Scotland

Using AmgX to accelerate a PETSc-based immersed-boundary method code

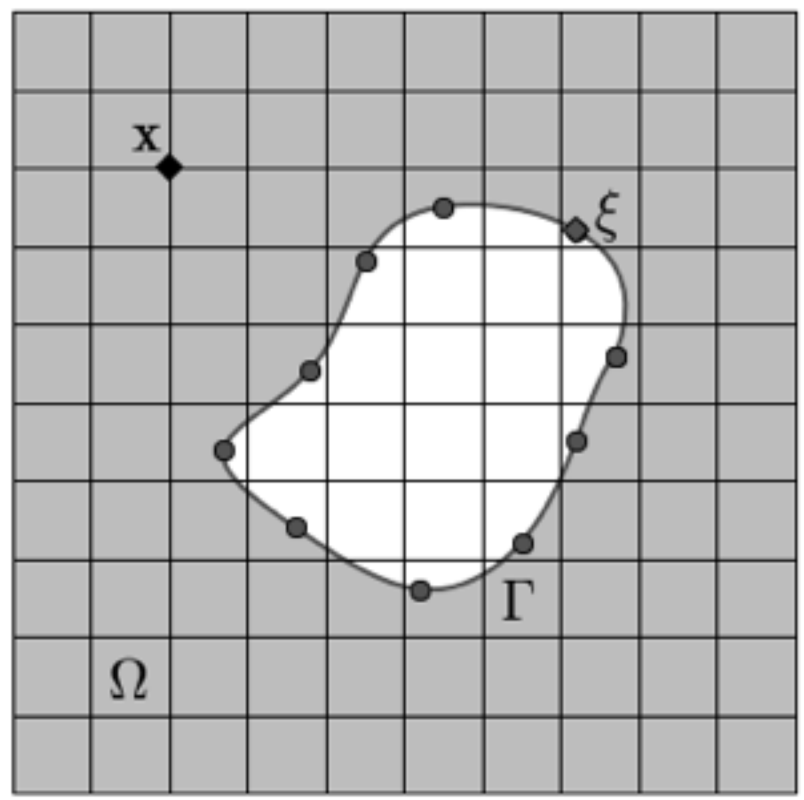
Olivier Mesnard, Pi-Yueh Chuang, & Lorena A. Barba

Mechanical and Aerospace Engineering,
The George Washington University, United-States

PetIBM

https://github.com/barbagroup/PetIBM

- 2D & 3D incompressible Navier-Stokes equations
- Projection method: a block-LU decomposition (Perot, 1993)
- PETSc (Portable Extensible Toolkit for Scientific Computation)
- Immersed-boundary method



$$\begin{cases} \frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = -\nabla p + \frac{1}{Re}\nabla^2 \mathbf{u} + \int_{\mathbf{s}} \mathbf{f}\left(\xi\left(s,t\right)\right)\delta\left(\xi - \mathbf{x}\right)\mathbf{d}s \\ \nabla \cdot \mathbf{u} = \mathbf{0} \\ \mathbf{u}\left(\xi\left(s,t\right)\right) = \int_{\mathbf{x}} \mathbf{u}\left(\mathbf{x}\right)\delta\left(\mathbf{x} - \xi\right)\mathbf{dx} \end{cases}$$

Immersed-boundary projection method (IBPM)

- Taira and Colonius (2007)
- Pressure and boundary forces gathered together
- Modified-Poisson system $Q^T B^N Q$

$$\begin{bmatrix} A & G & E^T \\ G^T & 0 & 0 \\ E & 0 & 0 \end{bmatrix} \begin{pmatrix} q^{n+1} \\ \phi \\ \tilde{f} \end{pmatrix} = \begin{pmatrix} r^n \\ 0 \\ u_B^{n+1} \end{pmatrix} + \begin{pmatrix} bc_1 \\ -bc_2 \\ 0 \end{pmatrix}$$

with $Q \equiv [G, E^T]$

$\lambda \equiv \begin{pmatrix} \phi \\ \tilde{f} \end{pmatrix}$

Block-LU decomposition:

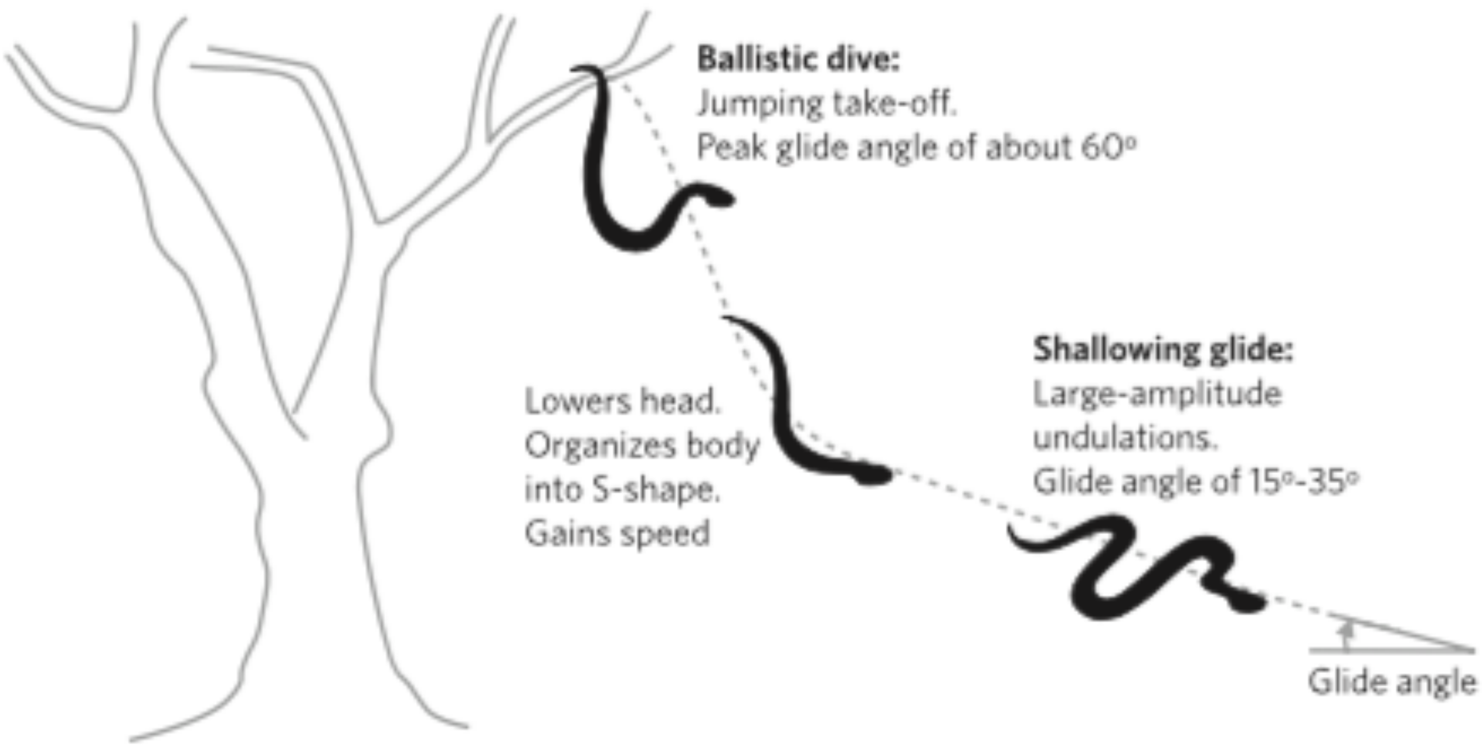
$$\begin{bmatrix} A & 0 \\ Q^T & -Q^T B^N Q \end{bmatrix} \begin{bmatrix} I & B^N Q \\ 0 & I \end{bmatrix} \begin{pmatrix} q^{n+1} \\ \lambda \end{pmatrix} = \begin{pmatrix} r_1 \\ r_2 \end{pmatrix}$$

$$Aq^* = r_1$$ **(velocity system)**

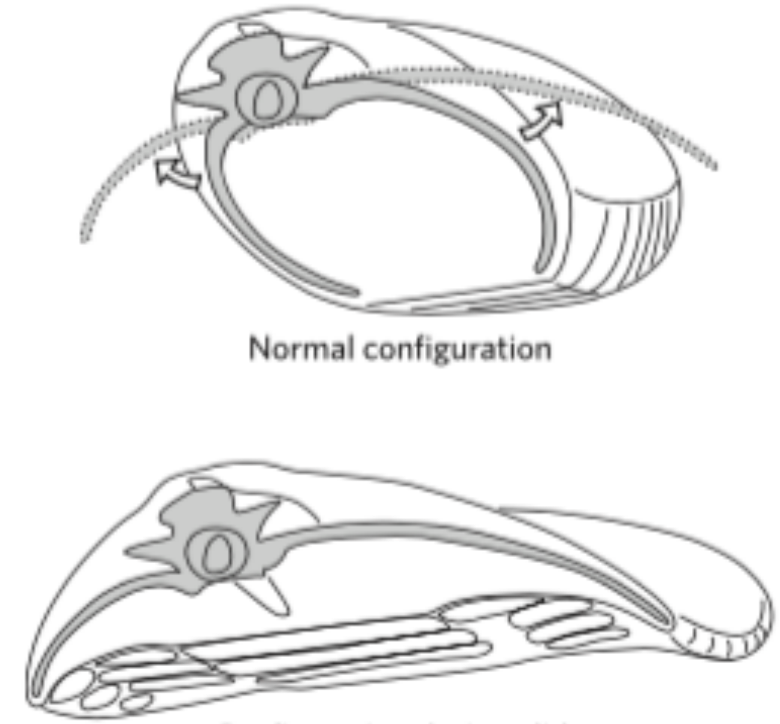
$$Q^T B^N Q\lambda = Q^T q^* - r_2$$ **(modified-Poisson system)**

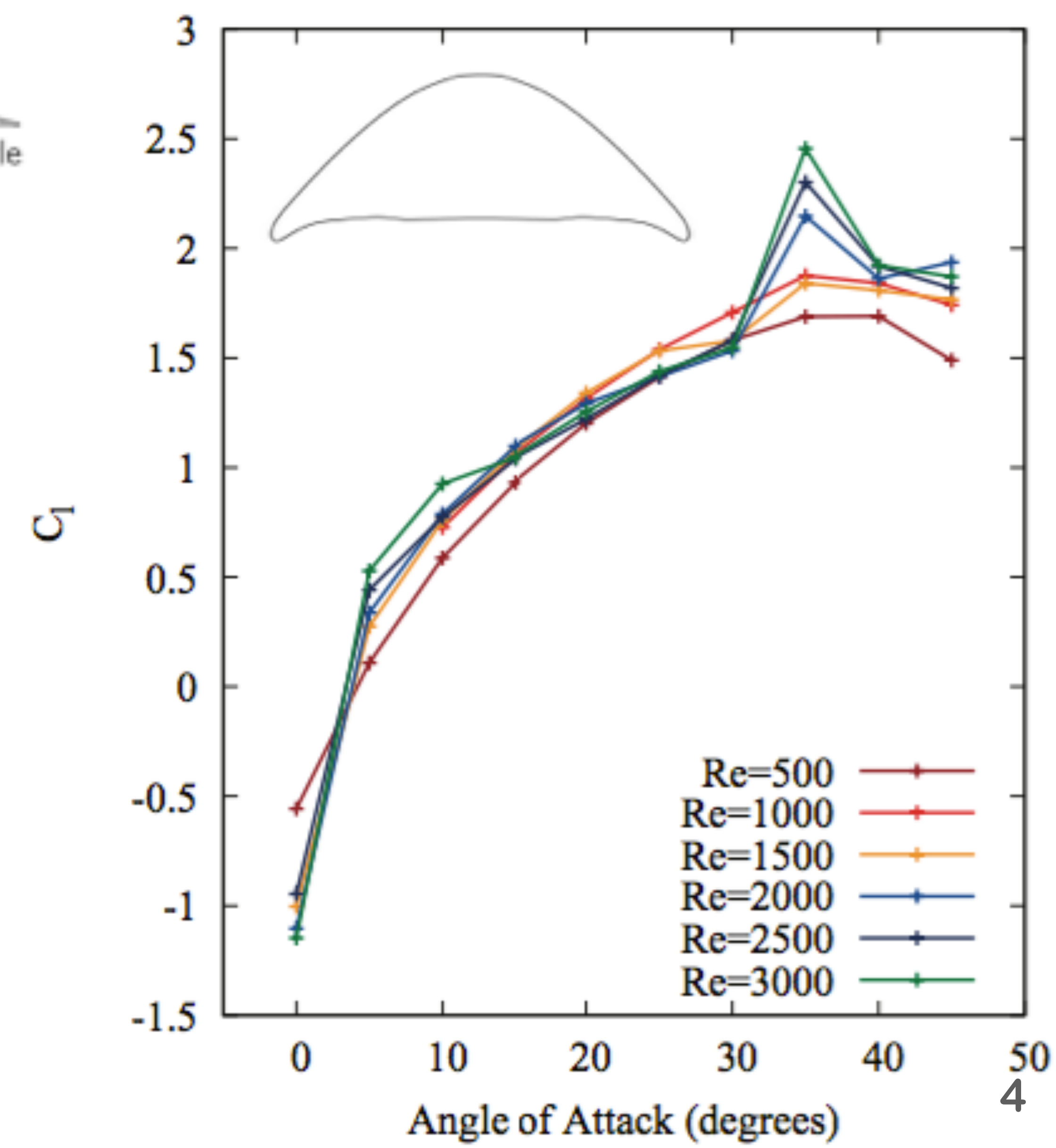
$$q^{n+1} = q^* - B^N Q\lambda$$ **(projection step)**

Application to 2D gliding snake



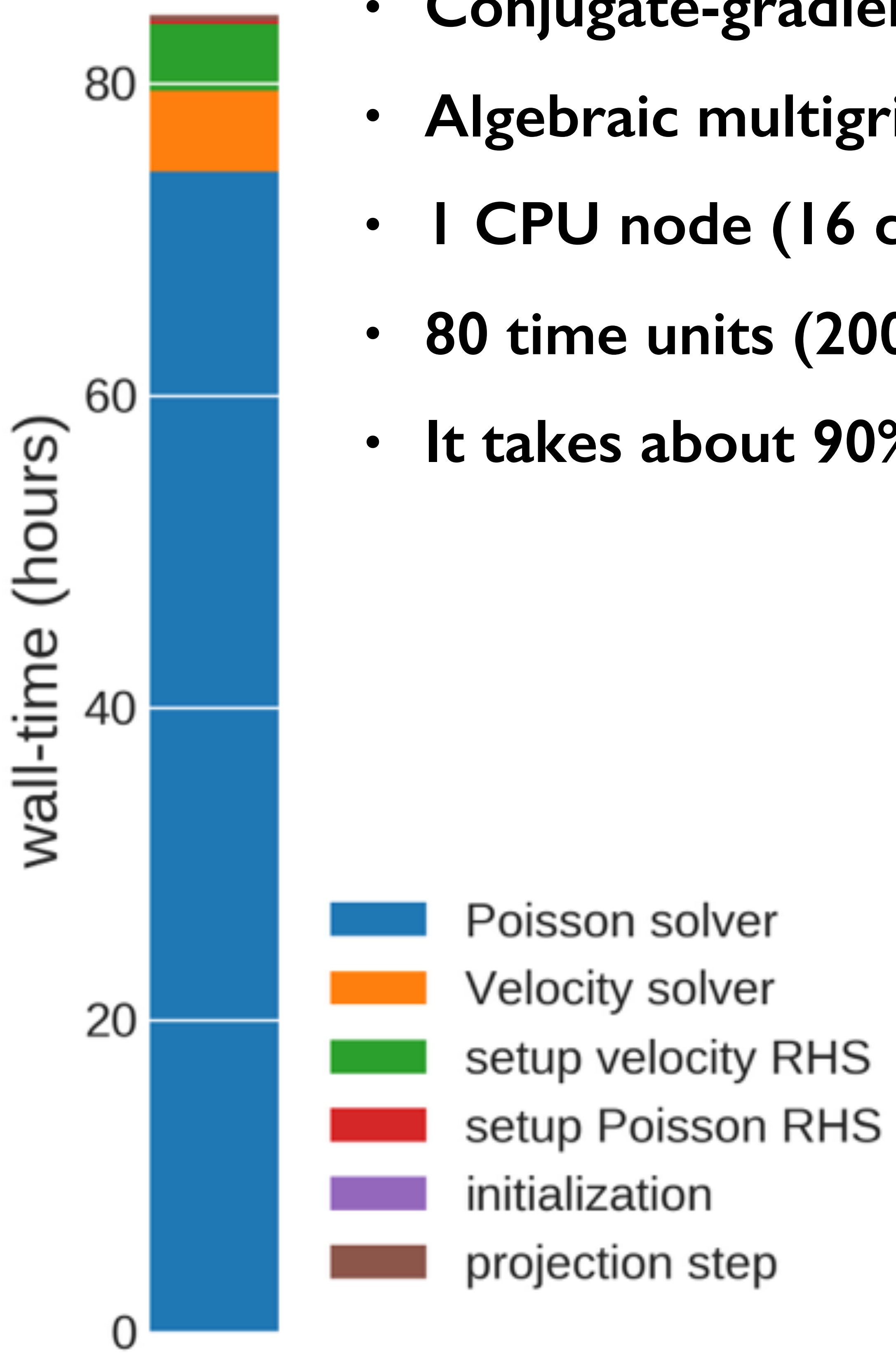
Figures from
Krishnan, A., Socha, J. J., Vlachos, P. P., & Barba, L. A. (2014).
Lift and wakes of flying snakes.
Physics of Fluids, 26(3), 031901.





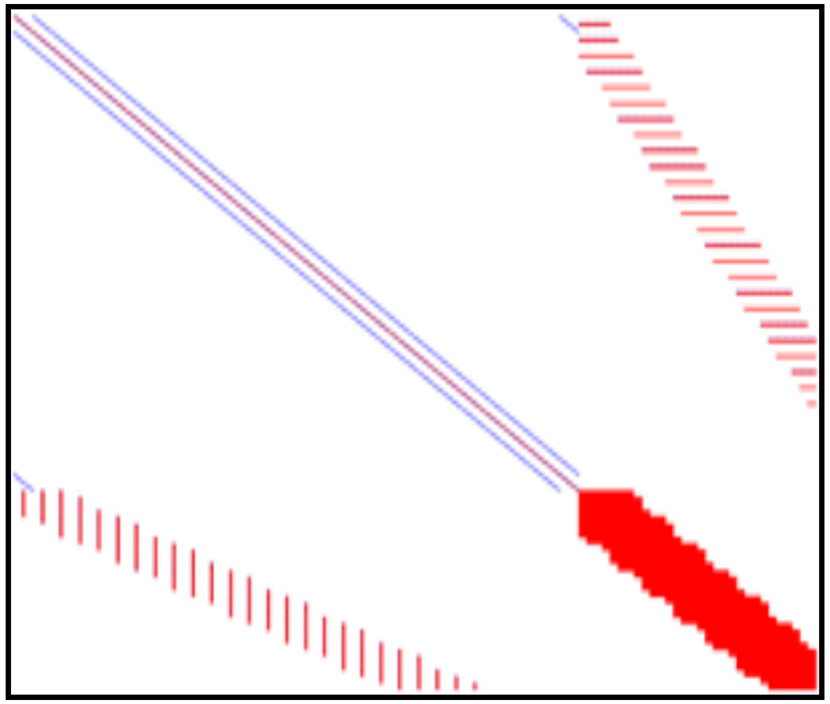
- Enhanced lift at Re ≥ 2000 at AoA 35 deg
- AoA matches previous experimental data
- Can we observe lift-enhancement with more realistic 3D simulations?

Modified-Poisson system expensive to solve



- Conjugate-gradient
- Algebraic multigrid preconditioner (aggregation)
- 1 CPU node (16 cores - Dual 8-Core 2.6GHz Intel Xeon E5-2670)
- 80 time units (200,000 time steps)
- It takes about 90% of the simulation runtime!

$$Q^T B^N Q$$



NVIDIA AMGX LIBRARY

https://developer.nvidia.com/amgx

- Various Krylov solvers
- Algebraic multigrid algorithms (classical and aggregation)
- Solve systems on multiple CUDA-capable GPU devices
- Available with a free license for non-commercial use

  for Accelerated Computing Developers

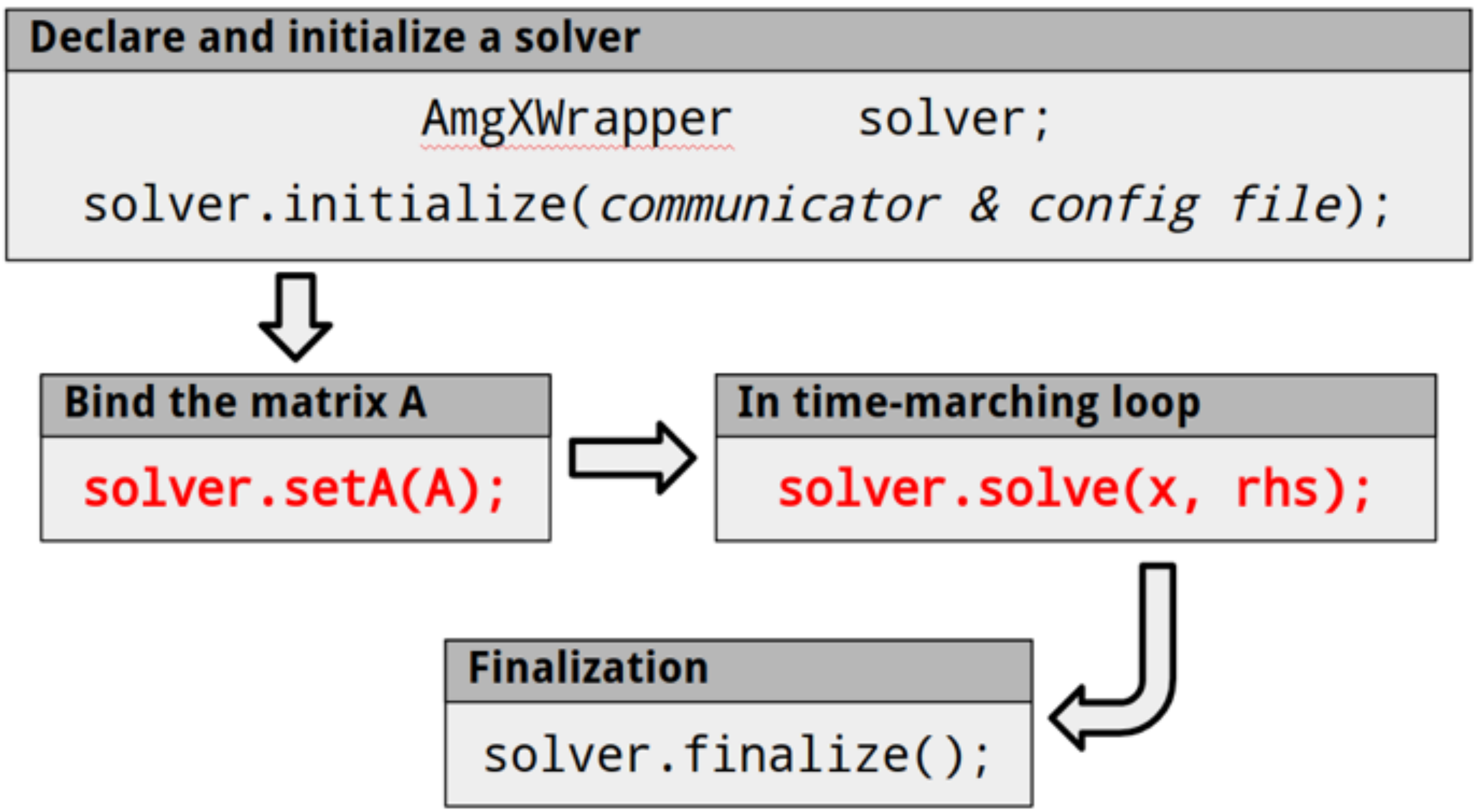
Objective: use AmgX within PetIBM to reduce the time-to-solution of the Poisson system

Problem: PETSc and AmgX have their own data structures

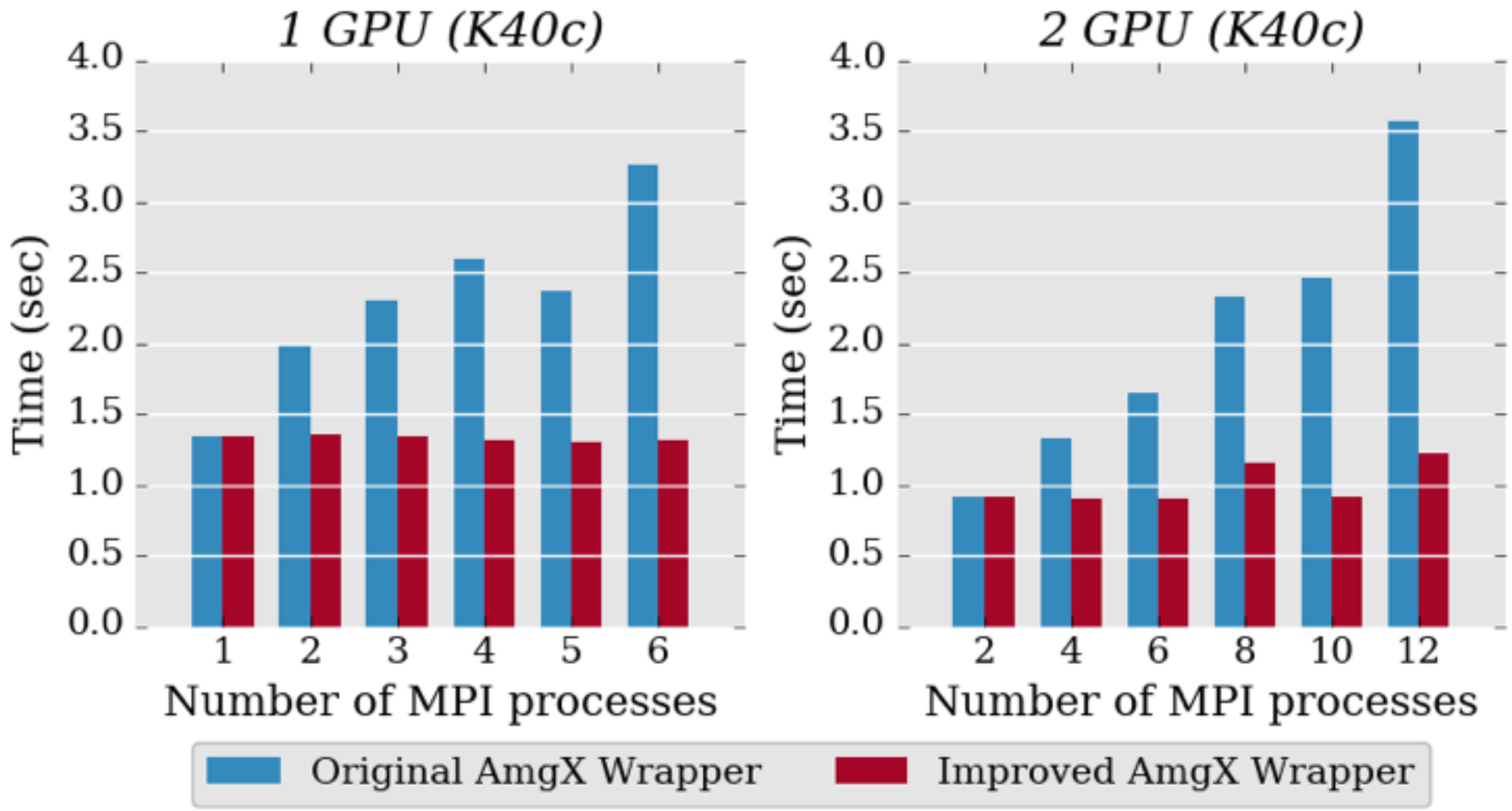
AmgXWrapper

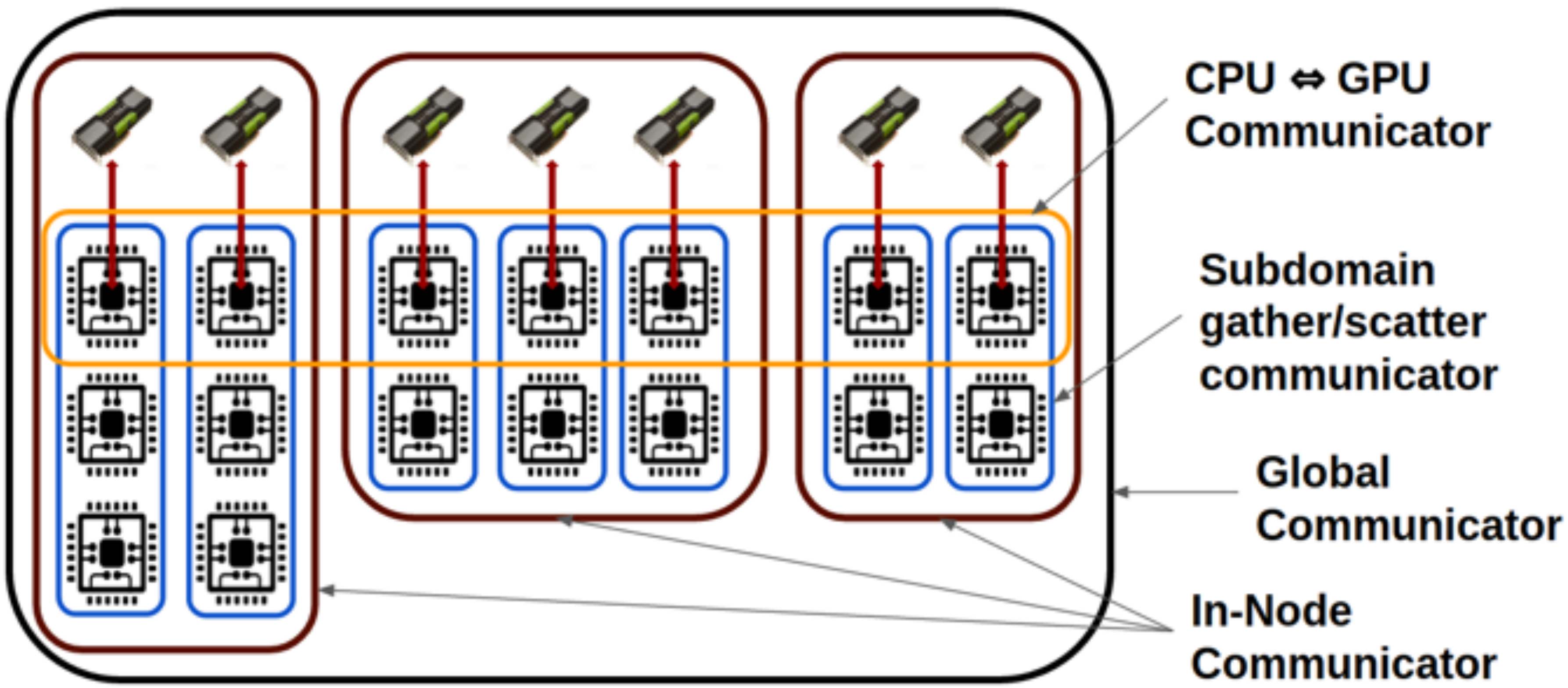
https://github.com/barbagroup/AmgXWrapper

- Interface between PETSc and AmgX
- Not specific to PetIBM



AmgXWrapper - Poisson system

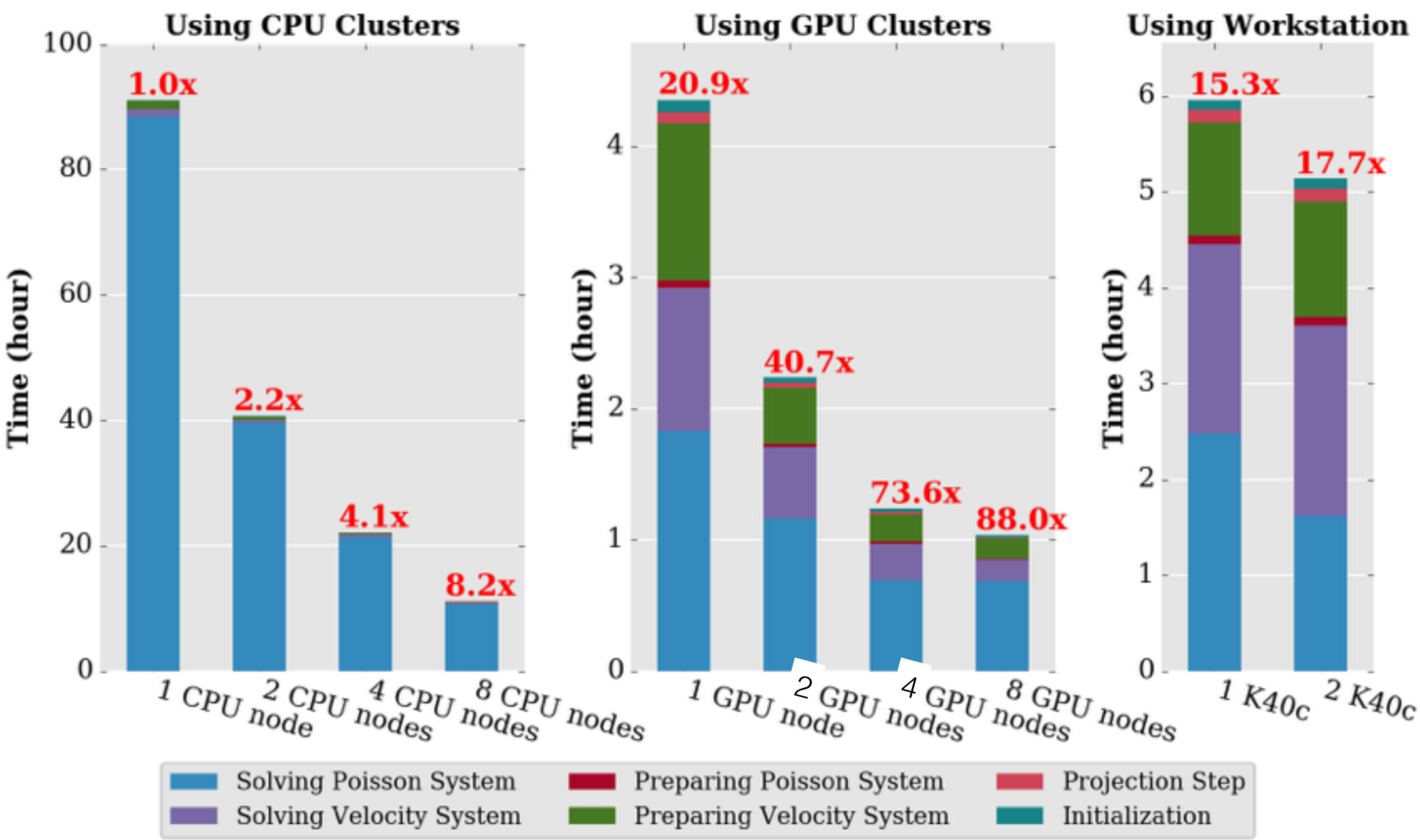




PetIBM + AmgXWrapper

- 1 CPU node: 12 CPU cores (2 Intel E5-2620)
- 1 GPU node: 1 CPU node (i.e., 12 CPU cores) + 2 K20 GPUs
- workstation: 6 CPU cores (1 Intel i7-5930K) + 2 K40c GPUs

Benchmark: 2D snake (Re=2000, AoA=35deg) 2.9M mesh-grid



https://github.com/barbagroup/conferences/blob/master/GTC2016/GTC2016_S6355.ipynb

Decoupled Immersed-boundary projection method

- Li and co-workers (2016)
- Decouple pressure field from Lagrangian forces
- 2-step block-LU decomposition

$$\begin{bmatrix} A & G & E^T \\ G^T & 0 & 0 \\ E & 0 & 0 \end{bmatrix} \begin{pmatrix} q^{n+1} \\ \phi \\ \tilde{f} \end{pmatrix} = \begin{pmatrix} r^n \\ 0 \\ u_B^{n+1} \end{pmatrix} + \begin{pmatrix} bc_1 \\ -bc_2 \\ 0 \end{pmatrix}$$

$$\begin{bmatrix} \bar{A} & \bar{E}^T \\ \bar{E} & 0 \end{bmatrix} \begin{pmatrix} \gamma^{n+1} \\ \tilde{f} \end{pmatrix} = \begin{pmatrix} \bar{r_1} \\ \bar{r_2} \end{pmatrix}$$

with $\bar{A} \equiv \begin{bmatrix} \bar{A} & \bar{E}^T \\ \bar{E} & 0 \end{bmatrix}; \; \gamma^{n+1} \equiv \begin{pmatrix} q^{n+1} \\ \phi \end{pmatrix}$

First block-LU decomposition:

$$\begin{bmatrix} \bar{A} & 0 \\ \bar{E} & -\bar{E}\bar{A}^{-1}\bar{E}^T \end{bmatrix} \begin{bmatrix} I & \bar{A}^{-1}\bar{E}^T \\ 0 & I \end{bmatrix} \begin{pmatrix} \gamma^{n+1} \\ \tilde{f} \end{pmatrix} = \begin{pmatrix} \bar{r_1} \\ \bar{r_2} \end{pmatrix}$$

Second block-LU decomposition:

$$\begin{bmatrix} A & 0 \\ G^T & -G^T A^{-1} G \end{bmatrix} \begin{bmatrix} I & A^{-1}G \\ 0 & I \end{bmatrix} \begin{pmatrix} q^* \\ \phi \end{pmatrix} = \begin{pmatrix} r_1 \\ r_2 \end{pmatrix}$$

$$Aq^{**} = r_1$$

$$G^T A^{-1} G\lambda = G^T q^{**} + bc_2$$

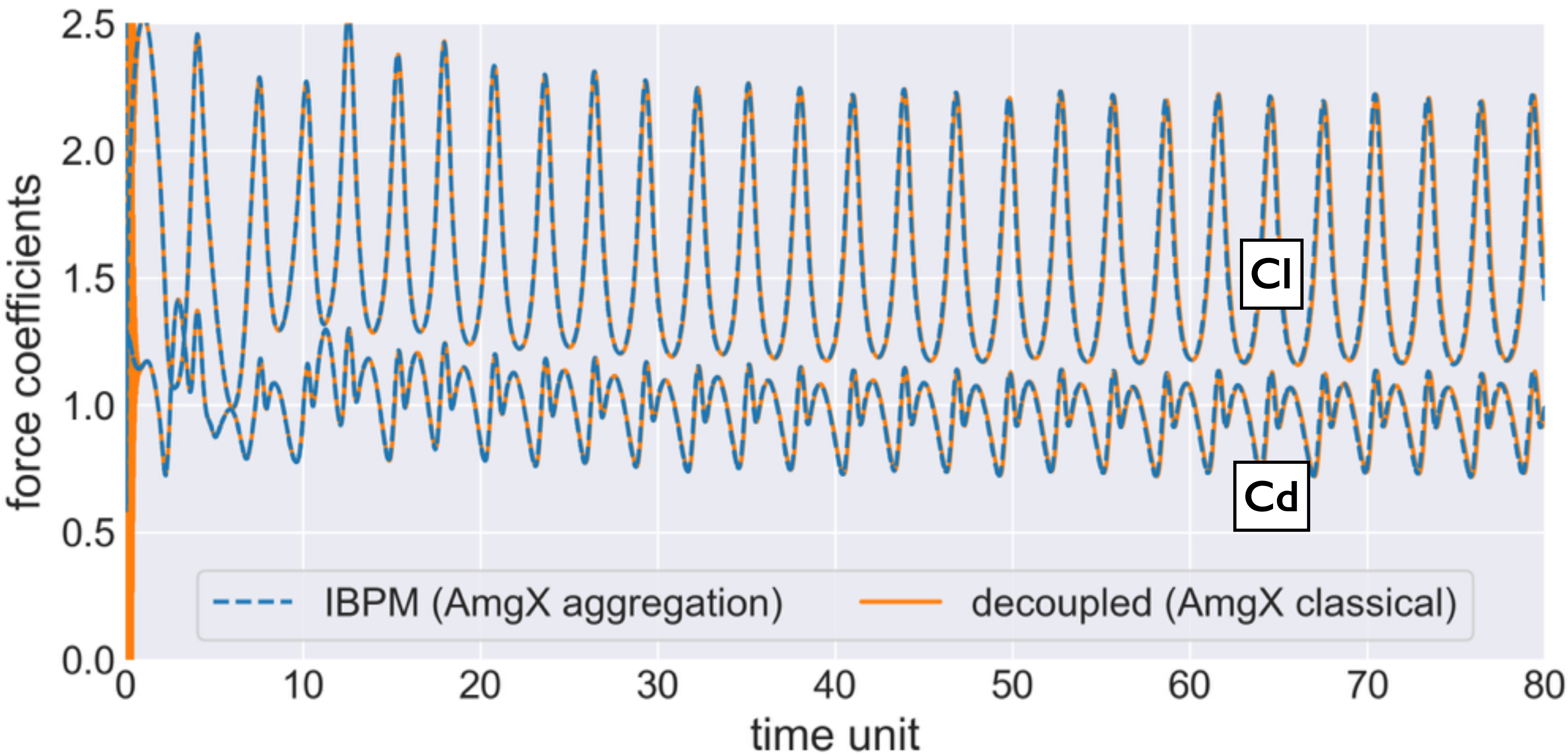
$$q^* = q^{**} - A^{-1}G\phi$$

$$EA^{-1}E^T\tilde{f} = Eq^* - u_B^{n+1}$$

$$q^{n+1} = q^* - A^{-1}E^T\tilde{f}$$

IBPM vs. decoupled method

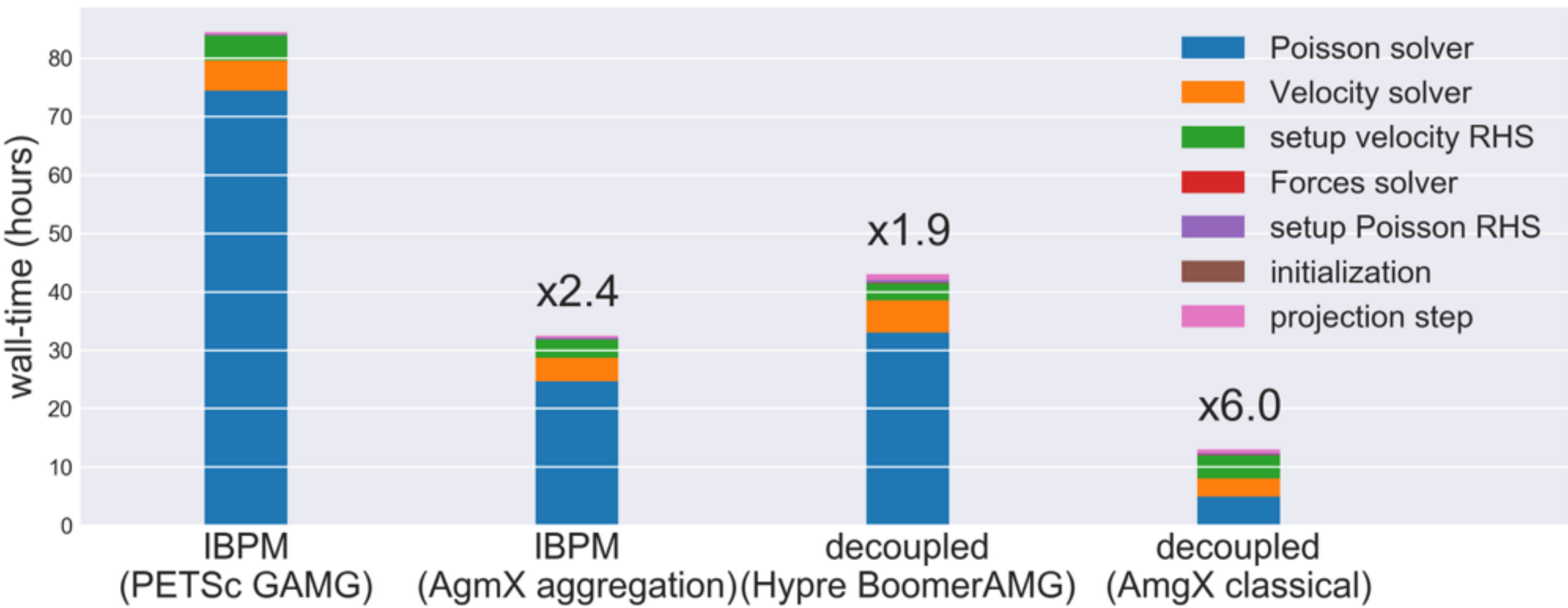
Benchmark: 2D snake (Re=2000, AoA=30deg)
2.9M meshgrid
200,000 time steps (80 time units)



IBPM vs. decoupled method

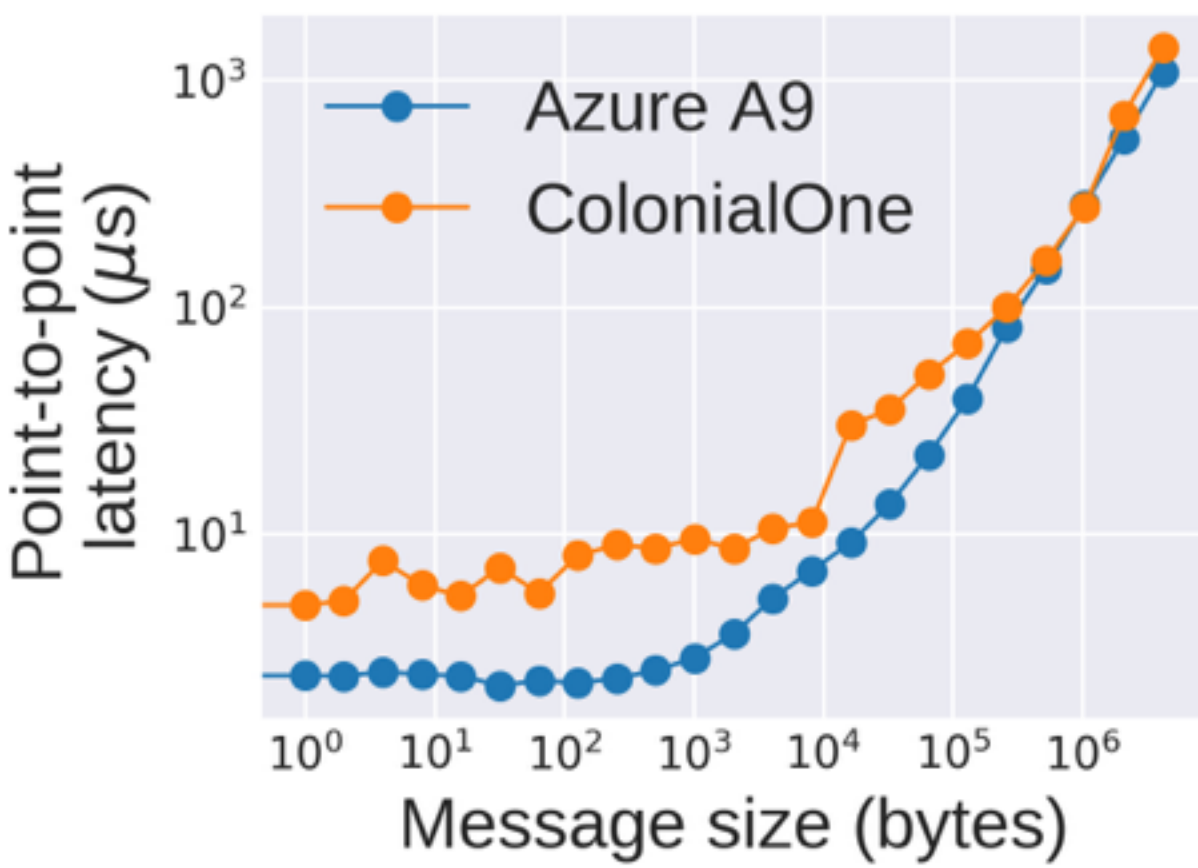
Benchmark: 2D snake
2.9M meshgrid
200,000 time steps (80 time units)

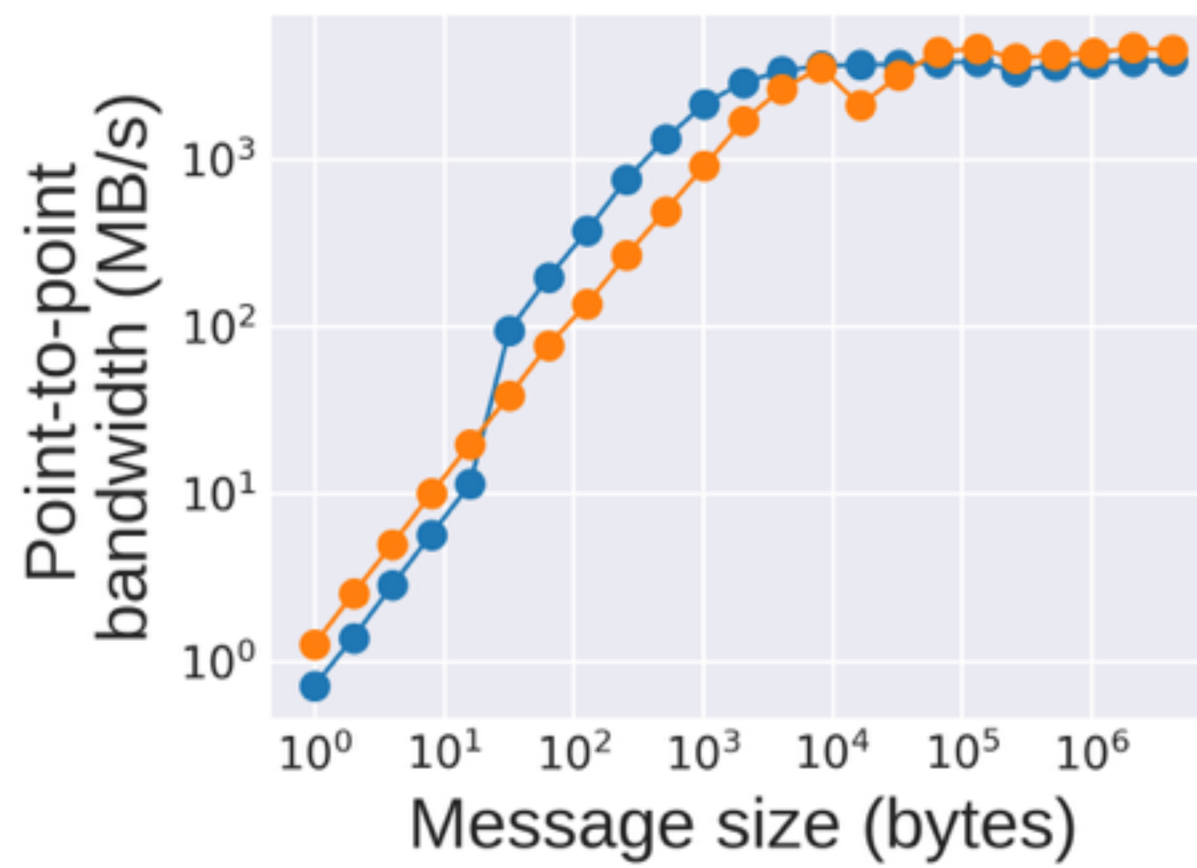
- 1 CPU node (16 CPU cores)
- 1 GPU node (12 CPU cores & 2 K20 devices)



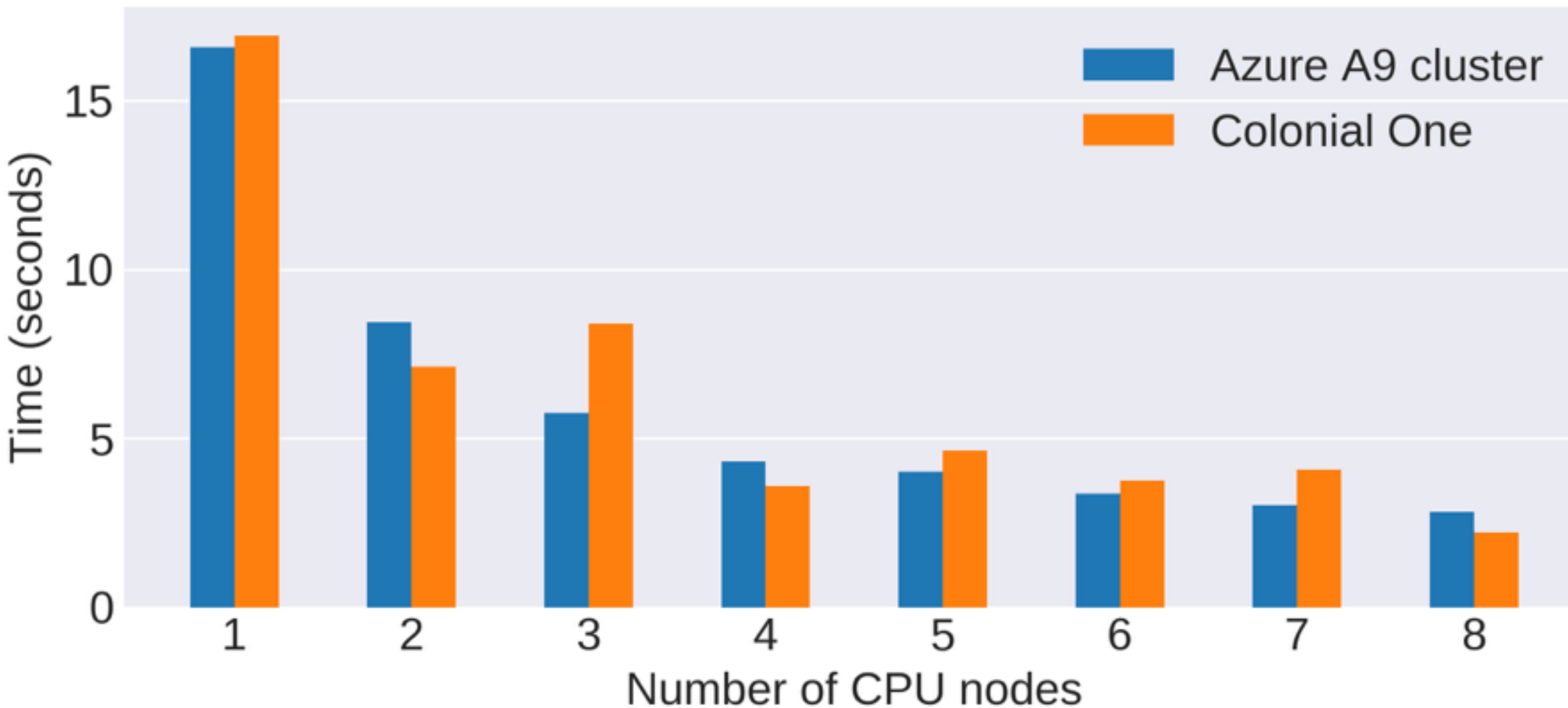
- CPU node on Colonial One
  - Dual 8-Core 2.6GHz Intel Xeon E5-2670 CPUs
  - InfiniBand
- Microsoft Azure A9
  - Dual 8-Core 2.6GHz Intel Xeon E5-2670 CPUs
  - InfiniBand

Ohio State University micro-benchmarks:





Benchmark: Poisson system with 46M unknowns,
Hypre BoomerAMG classical preconditioner
PETSc CG
16 CPU cores per node



Flying snakes to the cloud

A-series

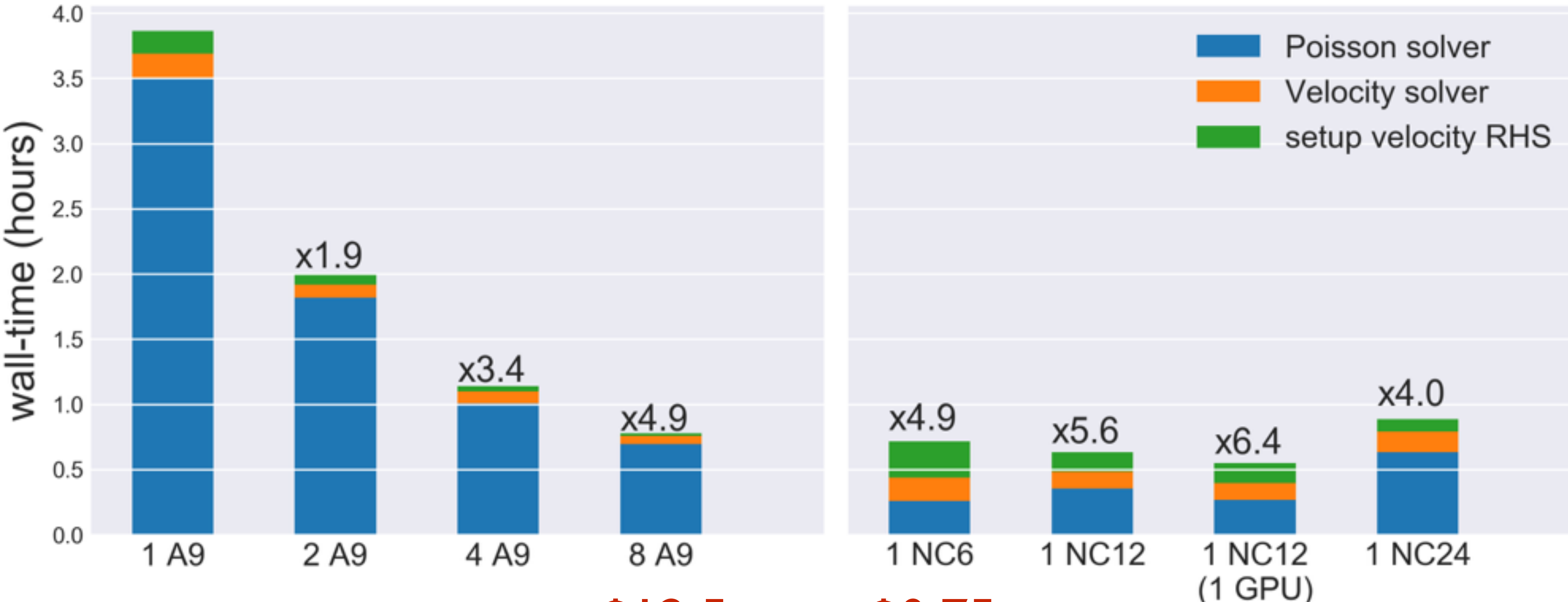
Instance	cores	RAM	disk sizes	price
A8	8	56GB	382GB	$0.975/hr
A9	16	112GB	382GB	$1.95/hr
A10	8	56GB	382GB	$0.78/hr
A11	16	112GB	382GB	$1.56/hr

NC-series

Instance	cores	RAM	disk sizes	GPU	price
NC6	6	56GB	340GB	1 x K80	$0.90/hr
NC12	12	112GB	680GB	2 x K80	$1.80/hr
NC24	24	224GB	1,440GB	4 x K80	$3.60/hr
NC24r	24	224GB	1,440GB	4 x K80	$3.96/hr

Flying snakes to the cloud

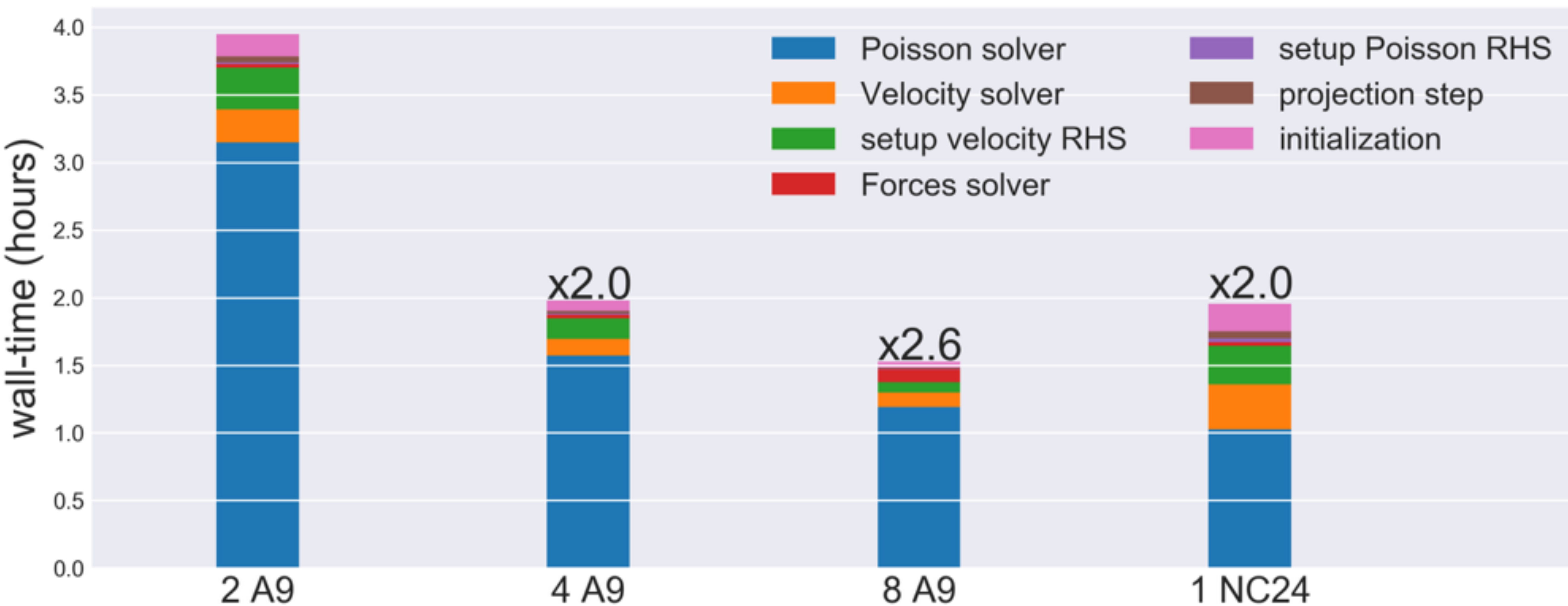
Benchmark: 2D snake (Re=2000, AoA=35deg)
2.9M meshgrid
10,000 time steps



$12.5 $0.75

Flying snakes to the cloud

Benchmark: 3D snake (Re=2000, AoA=deg)
46M meshgrid
1,000 time steps



$15.5 $7.9

Conclusions

- Use AmgX in a PETSc-based code
- AmgXWrapper (https://github.com/barbagroup/AmgXWrapper)
- Fast decoupled immersed-boundary projection method
- PetIBM + AmgX to reduce cloud computing expenses
- Microsoft Azure Sponsorship (www.azure4research.com)