

Non-blocking Preconditioned Conjugate Gradient Methods for Extreme-scale Computing

Paul Eller¹, William Gropp (Advisor)¹

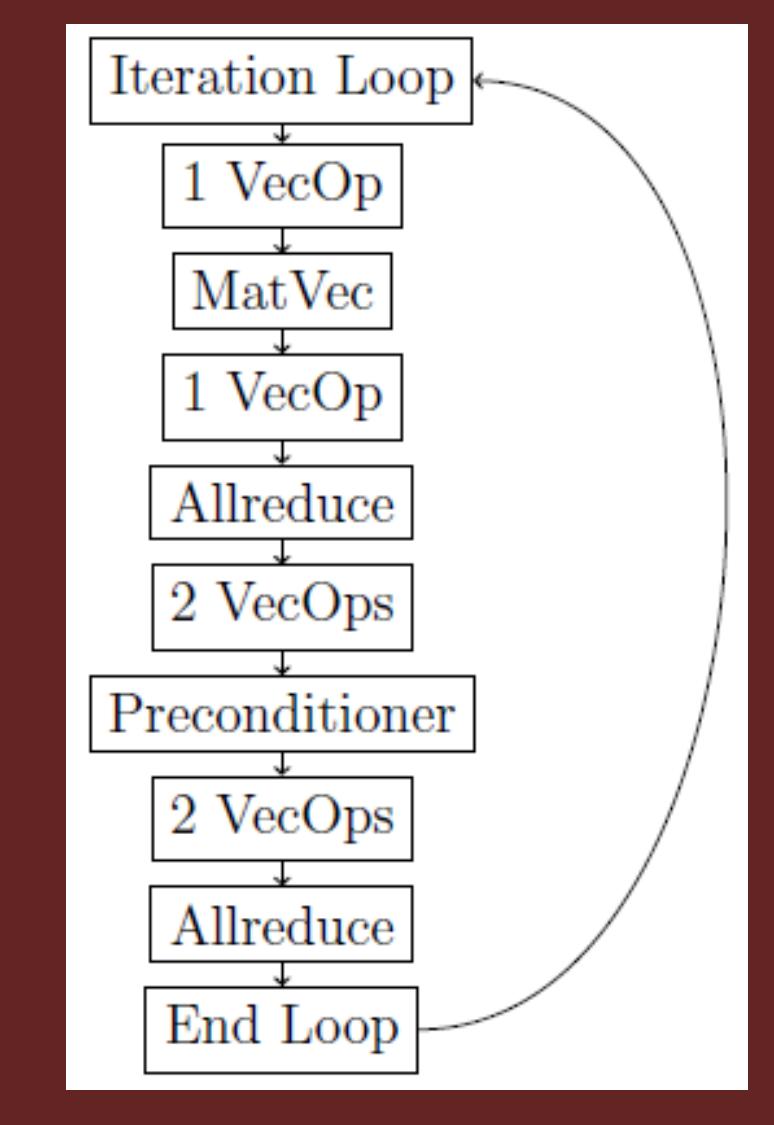
¹University of Illinois at Urbana-Champaign

Abstract

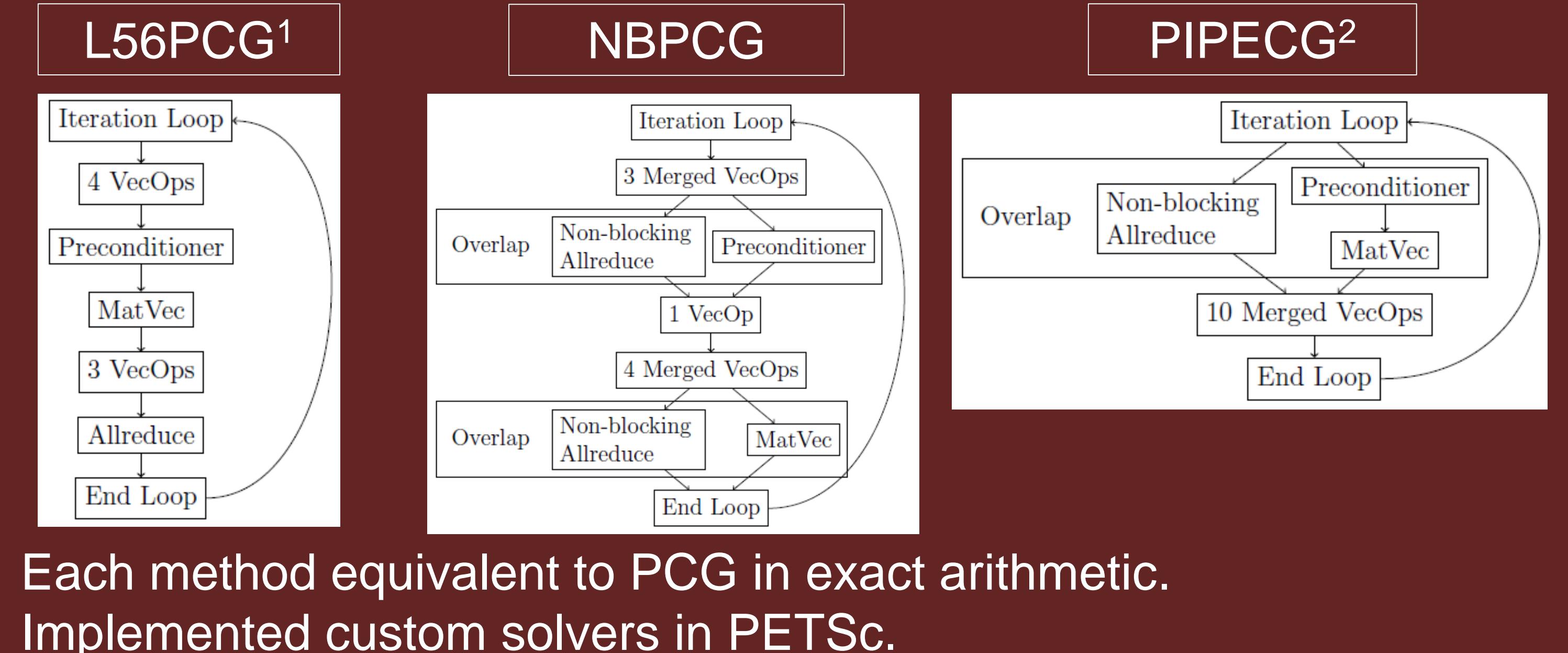
To achieve the best performance on extreme-scale systems we need to develop more scalable method variations. For PCG, dot products limit scalability because they are a synchronization point. Non-blocking methods provide potential to hide most of the cost of the allreduce and avoid synchronization cost due to performance variation across cores.

Preconditioned Conjugate Gradient Method (PCG)

- Iterative algorithm for solving large sparse systems of linear equations.
- Preconditioners accelerate convergence.
- Can rearrange PCG to:
 - Reduce communication latency using a single allreduce (L56PCG, PIPECG).
 - Overlap communication and computation using non-blocking allreduces (NBPCG, PIPECG).
- Optimizations introduce vector operations and initialization costs.



Scalable Conjugate Gradient Methods



- Each method equivalent to PCG in exact arithmetic.
- Implemented custom solvers in PETSc.

Non-blocking Allreduce

Current Approaches:

1. MPI_Test: MPI_Test calls during computation give MPI control of process so it can make progress on allreduce, but requires code modifications.

2. Progress Threads: Dedicate one or more threads per node to communication.

Future Approach:

3. Hardware Acceleration: Executes non-blocking allreduce in hardware, allowing processors to focus on computation.

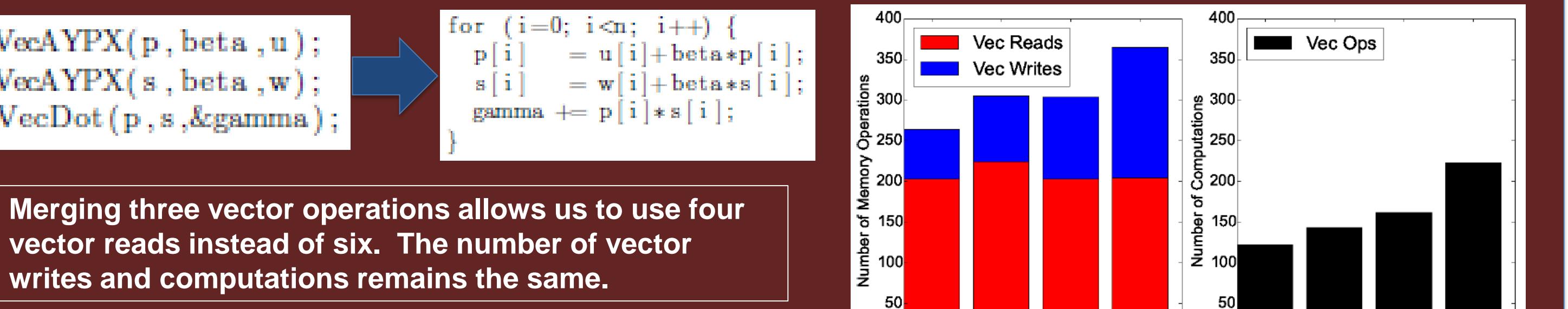
MPI_Test Approach

```
MPI_Iallreduce(...);
for (...) {
  do_chunk_computation();
  if (timetotest) {
    MPI_Test(...);
  }
  MPI_Wait(...);
}
```

Ideal Approach

```
MPI_Iallreduce(...);
do_all_computation();
MPI_Wait(...);
```

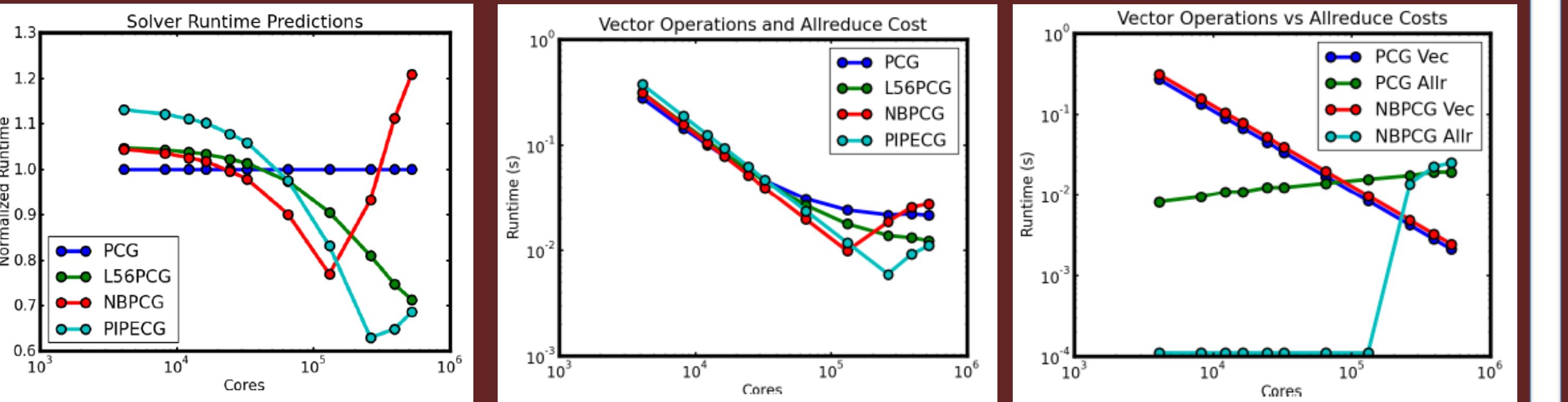
Merged Vector Operations



- Merging vector operations avoids cost of extra vector reads.
- Rearranged methods still require additional writes to memory.
- Requires additional computations, but these are cheap compared to memory accesses.

Performance Modeling

- Determine memory access and compute costs with modified STREAM benchmark.
- Model communication with LogGOPS.
- Compute parameters with Netgauge.
- Analyze performance using strong scaling tests.



General Observations

- Non-blocking methods perform better than blocking methods as the vector operations cost decreases and allreduce cost increases.
- Non-blocking methods perform well while the MatVec and/or PC have enough computation to hide the cost of the allreduce.
- NBPCG initially outperforms PIPECG due to lower vector operations cost.
- PIPECG scales better due to overlapping cost of allreduce with computation of both the MatVec and PC instead of only one.

Conclusions

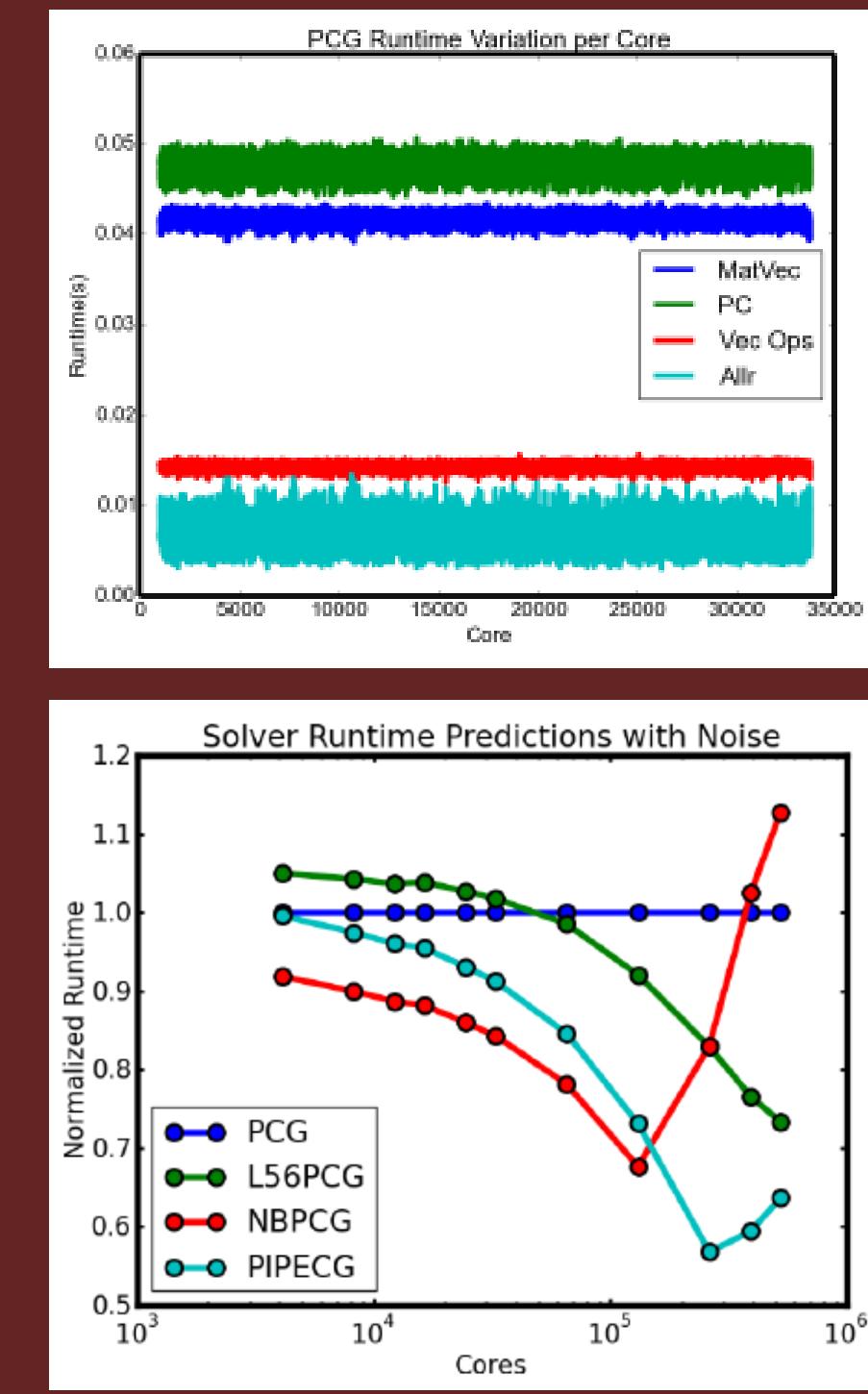
- Non-blocking solvers provide potential to improve performance at scale due to hiding cost of allreduce and avoiding synchronization.
- Current implementations cannot yet outperform standard PCG.
- Performance models show potential for NBPCG and PIPECG to be more scalable than PCG.
- Ability to minimize impact of noise may be key benefit.

Impact of Noise

Noise throughout PCG limits performance by causing all processes to wait for slowest process at sync points.

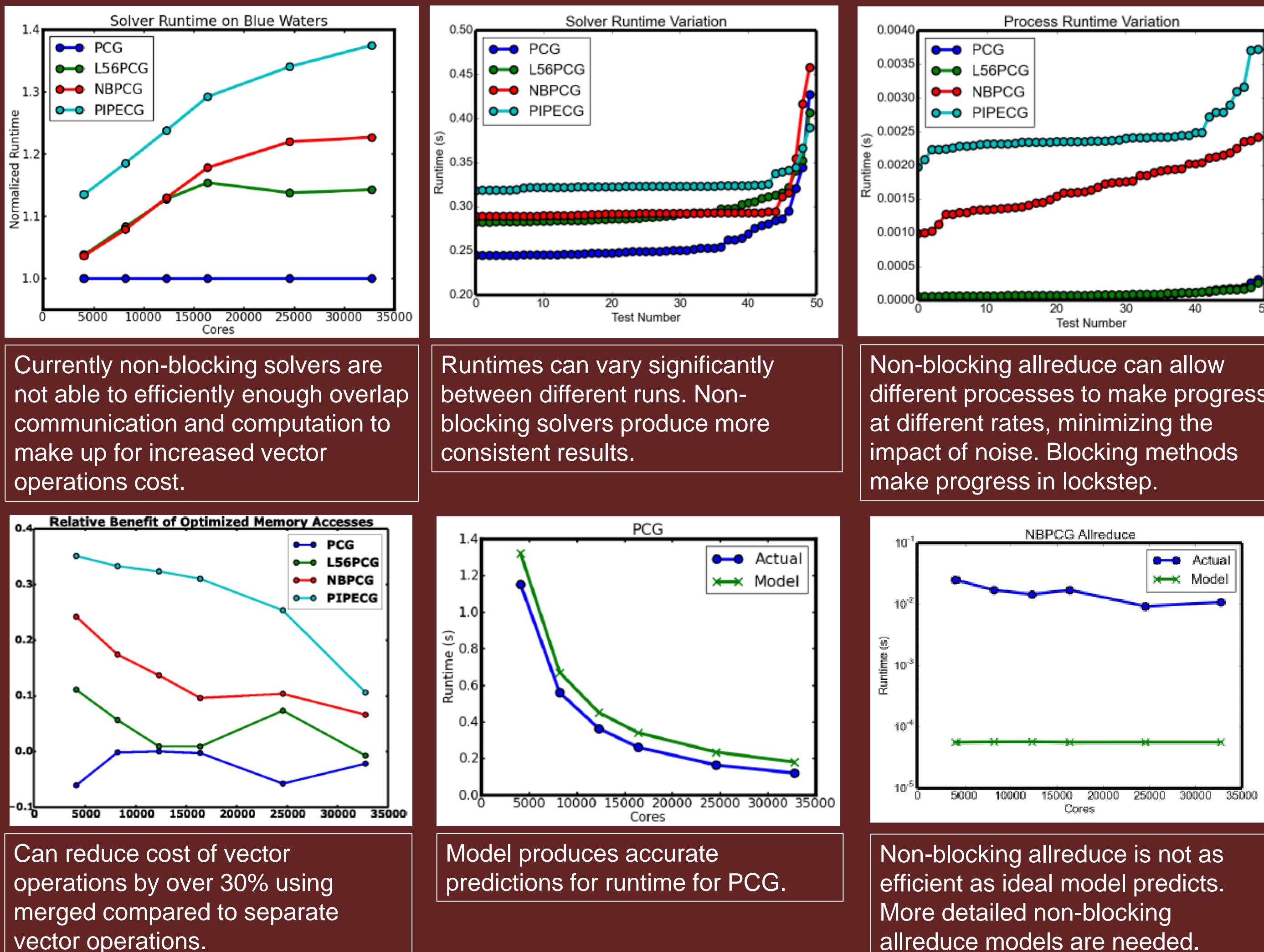
- Computational noise sources: Operating system processes, error correction, etc.
- Communication noise sources: Contention in network, varying distances between nodes, varying size/number of messages, etc.

Performance models predict ability to minimize impact of noise may be a key advantage to non-blocking solvers.



Solver Performance Results

- Tests are run on Blue Waters on up to 32k cores using 5-point 1 billion row Poisson matrices and Block Jacobi ILU preconditioner.



Currently non-blocking solvers are not able to efficiently enough overlap communication and computation to make up for increased vector operations cost.

Runtimes can vary significantly between different runs. Non-blocking solvers produce more consistent results.

Non-blocking allreduce can allow different processes to make progress at different rates, minimizing the impact of noise. Blocking methods make progress in lockstep.

- Can reduce cost of vector operations by over 30% using merged compared to separate vector operations.
- Model produces accurate predictions for runtime for PCG.
- Non-blocking allreduce is not as efficient as ideal model predicts. More detailed non-blocking allreduce models are needed.

¹E. D'Azevedo, V. Eijkhout, and C. Romine. Lapack working note 56: Reducing communication costs in the conjugate gradient algorithm on distributed memory multiprocessors. Technical Report, 1993.

²P. Ghysels and W. Vanroose. Hiding global synchronization latency in the preconditioned Conjugate Gradient algorithm. Journal of Parallel Computing, Vol 40, Issue 7, 2014.