# Optimising the High-Performance Linpack Benchmark on Two Different Clusters

MAISY DUNLAVY, University of New Mexico, USA
SUMAYA MOHAMED, University of New Mexico, USA
GERMAN BEJARANO, University of New Mexico, USA

ABSTRACT
In High-Performance Computing (HPC), optimising performance is crucial. High Performance Linpack (HPL) serves as a cornerstone metric, quantifying execution efficiency through Gflop/s. On Hopper, peak performance reached $2.03x10^3$ GFLOPS with a configuration of P=8 and Q=16 across four nodes, each accommodating 32 tasks. Wheeler cluster tests, initially conducted on two nodes, showcased performance enhancements with grid reconfigurationz and increased node counts. Subsequent block size benchmarks revealed optimal performance with a block size of 32. Further tests explored grid configurations, problem size optimisations, and memory constraints, achieving $4.0157x10^3$ GFLOPS with a problem size of 334500. These findings show the correlation between computational parameters and performance outcomes in HPC environments.

**ACM Reference Format:**
Maisy Dunlavy, Sumaya Mohamed, and German Bejarano. 2024. Optimising the High-Performance Linpack Benchmark on Two Different Clusters. 1, 1 (December 2024), 4 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

## INTRODUCTION

ABSTRACT In the realm of High-Performance Computing (HPC), optimising performance stands as a fundamental concern. The way computational tasks are distributed across CPUs and compute nodes plays an important role in determining the efficiency of the applications being executed. At the heart of performance evaluation lies High Performance Linpack (HPL), a metric that measures the execution rate of solving a linear equation. Using MPI (Message Passing Interface) and specialised linear algebra libraries, HPL quantifies performance in terms of Gflop/s, denoting the rate at which the machine executes 64-bit floating-point operations per second. The Theoretical performance of a machine is determined by the number of additions and multiplications that can be completed in the cycle time of the machine rather than bench marking, thus setting a theoretical upper limit on achievable performance.

## METHODS

HOPPER - In the investigation of block size variations ranging from 32 to 1024, the anticipated peak performance was approximately 300. Surprisingly, the observed highest performance was found near 1000. Increasing the block size for smaller numbers ($< 200$) to higher numbers shows the largest increase we see on the graph, then the GLFOPS seem to mostly stabilise at around 300 NB. The block size shows some drops and spikes that are not super intuitive. This could be caused by a multitude of reasons, most prominently the resources available on the cluster at the time I was running my jobs.

The machines all of these tests were run on are open to many researchers, students and professors at UNM. Due to this, I was not able to specify specific hardware that I wanted to run on and started off using less than the maximum amount of resources on the nodes I was running on. This introduces the possibility that some runs may end up on nodes next to each other, increasing communication speed and that I could be the only person running on the node, reducing communication overhead. On the other side of the spectrum, I could be running on the two nodes that are furthest away from each other, and other jobs are running on the other CPU's on the machine when my job runs. There isn't a great way to know which of these scenarios (or what combination of the two) played out without actually watching the job get allocated, sshing onto the node and watching HTOP. I queued all my jobs, so this information isn't readily available. This likely explains a good amount of variation in the graph that does not seem to follow a specific pattern.

Fig. 2. Sensitivity to initial conditions

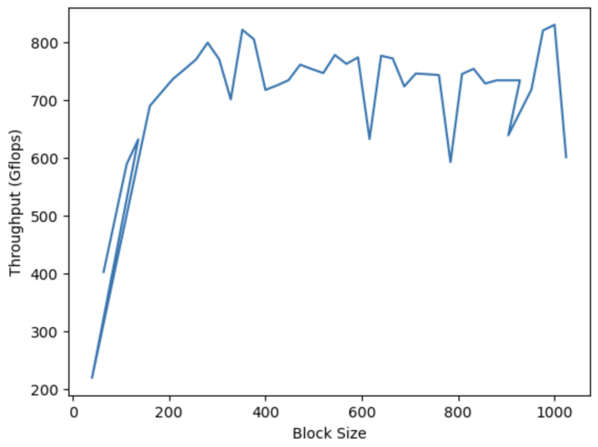
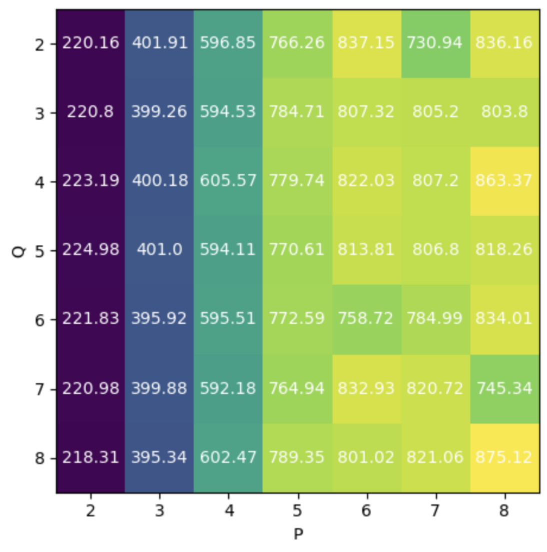


Figure 1: Throughput as a function of block size.

(a) This figure shows the scaling study for NB.

Regarding the ratio of processor grid dimensions (P:Q), an equitable distribution was initially expected due to the utilisation of

Authors' addresses: Maisy Dunlavy, University of New Mexico, Albuquerque, NM, 87131, USA; Sumaya Mohamed, University of New Mexico, Albuquerque, NM, 87131, USA; German Bejarano, University of New Mexico, Albuquerque, NM, 87131, USA.



| Q \ P | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|
| 2 | 220.16 | 401.91 | 596.85 | 766.26 | 837.15 | 730.94 | 836.16 |
| 3 | 220.8 | 399.26 | 594.53 | 784.71 | 807.32 | 805.2 | 803.8 |
| 4 | 223.19 | 400.18 | 605.57 | 779.74 | 822.03 | 807.2 | 863.37 |
| 5 | 224.98 | 401.0 | 594.11 | 770.61 | 813.81 | 806.8 | 818.26 |
| 6 | 221.83 | 395.92 | 595.51 | 772.59 | 758.72 | 784.99 | 834.01 |
| 7 | 220.98 | 399.88 | 592.18 | 764.94 | 832.93 | 820.72 | 745.34 |
| 8 | 218.31 | 395.34 | 602.47 | 789.35 | 801.02 | 821.06 | 875.12 |

(a) This figure illustrates the scaling study conducted for the parameters P and Q on the Hopper system. The tests were executed across two nodes. The variable "ntasks" was adjusted accordingly to ensure that the total number of tasks equalled the product of P and Q, maintaining a balanced distribution of computational workload across the specified number of processes.

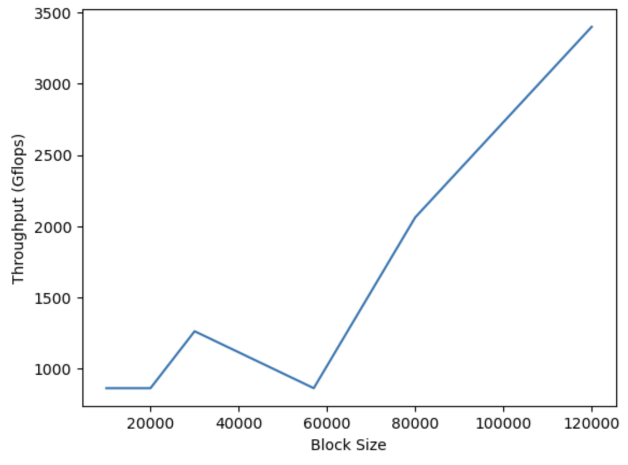


Figure 3: Throughput as a function of problem size.

(b) This figure represents the scaling study for N (problem size) for Hopper.

InfiniBand technology on the Hopper system, suggesting minimal communication. It was expected that under slower communication conditions, a flatter distribution of P and Q would yield optimal results, whereas in instances of faster communication, a more square configuration would be preferable. I chose to use a heat map because of previously tracking P and Q values for HPL scaling.

When I originally started my P:Q scaling study, I attempted to scale up and try to start my P and Q at 4 and 4 respectively. My jobs ended up taking long enough that I decided to scale down to 2 nodes rather than 4 and start from a P and Q of 2 and 2. This scaling study allowed me to get resources much faster on the debug queue and complete a more well rounded study. We see a similar amount of irregularity that we saw on block size for P and Q scaling. There are some P and Q values that are either significantly lower or higher than its neighbours. It is interesting to note that smaller P values are consistently lower performing regardless of Q values. This is not what I would expect. I would expect more of a diagonally increasing pattern rather than a vertically increasing one on the heat map. I am still looking into why P would have this pattern but Q would not show as strong of a correlation.

Furthermore, experiments involving the scaling of problem sizes revealed a direct correlation between increased problem size (N values) and GFLOPS performance. Although efforts to scale to higher magnitudes were undertaken to detect potential memory leaks, such occurrences were not observed. Instead, evidence of scalability was noted.

WHEELER - Baseline Configuration. Initial tests were conducted using default settings to establish a baseline performance. This baseline was used as a reference for all subsequent optimisations. To ensure using the full capacity of the cluster, the number of nodes used was increased from the default setting of 2 to 50. This adjustment necessitated a reconfiguration of the process grid, setting $P$ and $Q$ such that their product equaled the total number of processors (400). This configuration was maintained throughout all further tests.

Block Size. The block size (NB) was varied from 8 to 512 to determine the most effective size for computational efficiency. The optimal block size observed was then used in subsequent tests.

Grid Configuration. With the best block size fixed, various $P \times Q$ combinations that factored into 400 were systematically tested. This step was useful to understand the impact of process grid shape on performance, observing configurations from elongated (e.g., $1 \times 400$) to square-like (e.g., $20 \times 20$).

Problem Size. After determining the most effective block size and grid configuration, the problem size was varied. Starting from the default size, it was incrementally increased to determine the largest size that could be processed without exceeding the memory constraints of the system. Each increment was tested under the configuration that previously yielded the best performance results.

## RESULTS

### HOPPER

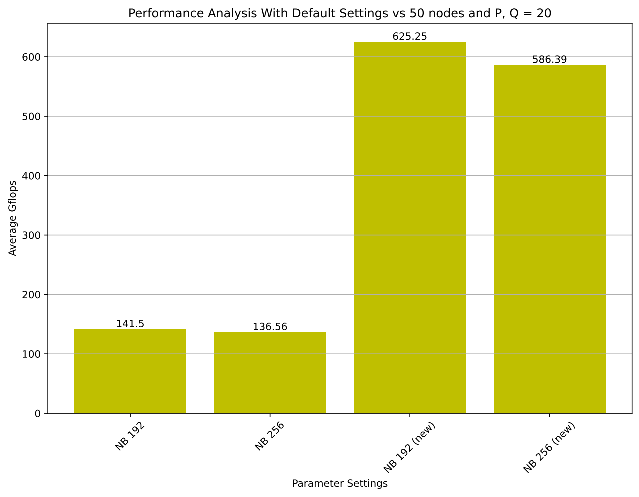
The highest recorded performance on the Hopper system reached

$2.03x10^3$ GFLOPS. This score was achieved by configuring the parallel processing parameters as follows: P set to 8 and Q to 16, totaling 128 computational tasks. The workload was distributed across four nodes, with each node accommodating 32 tasks. This configuration enabled the utilization of all four nodes, maximizing computational resources.

WHEELER

The initial benchmarks were conducted on the Wheeler cluster using two nodes, as opposed to the typical setup of 50 nodes with 8 processors each. The baseline parameters were as follows: BLOCK SIZE set to 192 and 256, PROBLEM SIZE at 14616, and a GRID configuration of P=2, Q=8. Subsequent tests utilized the full 50-node configuration, with an initial test repeating the baseline parameters but modifying the grid to 20x20. These changes resulted in a performance increase, as depicted in the bar graph in figure 4a. However, the specific contributions of the increased node count and grid reconfiguration to performance improvements could not be isolated.

Figure 4a.



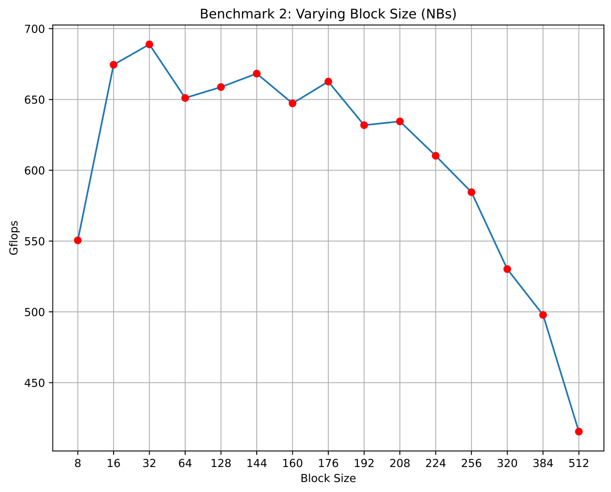
(a) Wheeler Baseline

In the subsequent block size benchmark, tests ranged from block sizes of 8 to 512, maintaining the problem size at 14616 and a grid configuration of 20x20. figure 4b illustrates these results, highlighting a block size of 32 as optimal under the given conditions.

Building on these findings, the following tests employed configurations that previously yielded the best performance. Specifically, the tests explored grid configurations of 1x400, 2x200, up through 20x20, including their inverses, with a fixed block size of 32 and problem size of 14616. The grid configuration of 5x80 emerged as the most effective, as shown in figure 4c.

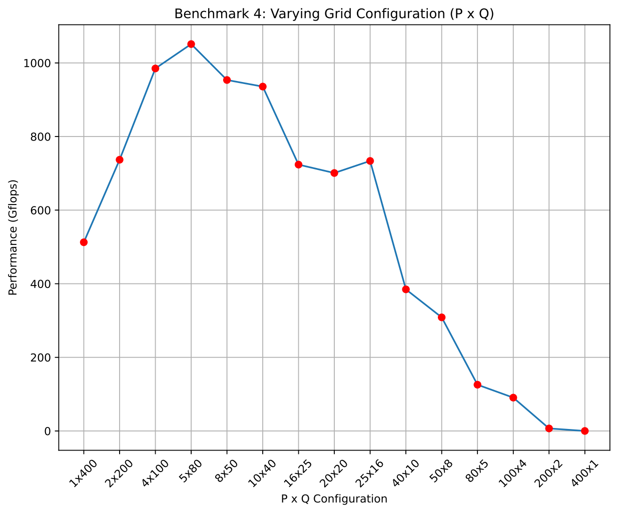
Further tests aimed to optimize problem size, incrementing from 14500 to 98500 in steps of 6000, using a block size of 32 and a grid of 5x80. These results, presented in figure 4d, indicated a positive correlation between larger problem sizes and enhanced performance. Due to time constraints, testing was capped at a problem size of

Figure 4b.



(b) Wheeler Block Size

Figure 4c.

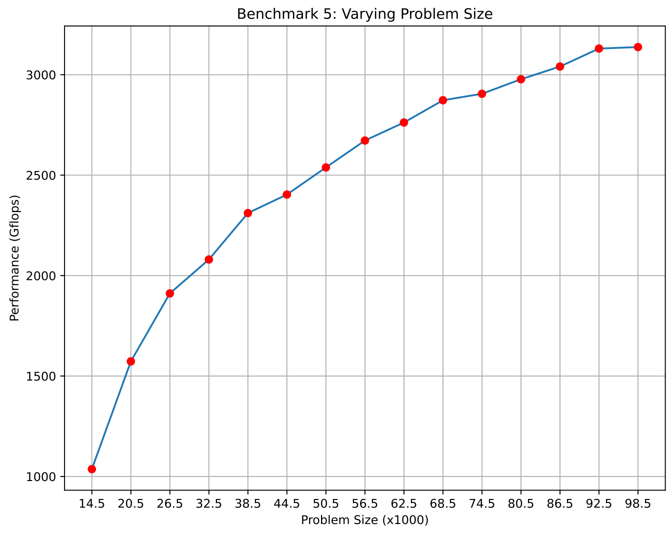


(c) Wheeler Grid Configuration

98500, although further exploration was conducted to identify the peak performance capacity.

To estimate the maximum feasible problem size, calculations based on the cluster's total memory were performed: 50 nodes $\times$ 44 GB $=$ 2200 GB, converting to bytes gives $2.36 \times 10^{12}$. Taking the square root of $\frac{2.36\times10^{12}}{8}$ and adjusting for 80% to accommodate system overhead resulted in an estimated 434716.6 elements.

Figure 4d.



(d) Wheeler Problem Size

In practice, the largest tested problem size was 334500, with a block size of 176 and a grid configuration of 20x20, achieving a performance of $4.0157 \times 10^3$ Gflops.

## CONCLUSIONS

The benchmark testing on the Wheeler cluster has demonstrated substantial performance improvements through systematic parameter fine tuning. Specifically, increasing the node count and optimizing the grid configuration from P=2, Q=8 to P=20, Q=20 significantly enhanced computational efficiency. The block size of 32 was identified as optimal for smaller problem sizes within the testing range, however, larger problem sizes benefited from larger block sizes, with 172 being best overall. The optimal settings identified, particularly the block size and grid configuration, suggest that similar large-scale computational environments could benefit from a fine tuning approach to configuration settings. These adjustments are crucial for maximizing performance on specific operations and could serve as a benchmark for future setups in similar high-performance computing setups.

Further research should explore even larger problem sizes and more diverse grid configurations, especially those that push the boundaries of current hardware capabilities. Additionally, exploring the impact of newer hardware technologies and different types of computational tasks could provide deeper insights into the scalability and adaptability of our findings. In conclusion, the fine tuning of computational parameters on the Wheeler cluster not only showed increased performance but also highlighted the essential role of system analysis in achieving superior computational performance.

This study highlights the potential for significant performance gains through methodical testing and adjustments, and demonstrating how parallelized computations can yield higher performance. The investigation on Hopper explored variations in block size, the highest performance was observed near 1000 GFLOPS. Notably, increasing the block size from smaller numbers (< 200) to higher ones showed the most substantial performance increase on the graph, stabilizing around 300 GFLOPS thereafter. However, fluctuations in performance were noted, attributed to factors such as resource availability and communication speed across the cluster. In the context of InfiniBand technology on Hopper, communication was expected to be minimal, influencing the choice of parallel processing configurations. The achieved peak performance of $2.03x10^3$ GFLOPS was realized with a configuration of P=8 and Q=16 across four nodes, with each node accommodating 32 tasks, effectively maximizing computational resources.

## AUTHOR CONTRIBUTIONS

{Maisy Dunlavy} Did all Hopper runs, first optimizing the Block Size (NB), then finding the best P:Q combination, then the best N value. Made graphs/heat map for hopper runs, wrote up notes and summaries for hopper runs.
{Sumaya Mohamed} worked on writing the report, including the abstract, the introduction, methods, results, and the conclusion.
{German Bejarano} Conducted computational experiments on the Wheeler cluster, focusing on evaluating the impact of varying parameters on performance. Responsible for all data collection from these tests. Utilised this data to create graphical representations, illustrating findings and trends. Authored the methods section detailing the experimental procedures on Wheeler.

## REFERENCES

Dongarra, J., Luszczek, P., Petitet, A. (2003). LINPACK Benchmarks on Heterogeneous Computers.
Netlib. Algorithmic aspects of the HPL Benchmark.
Dongarra, J. J. (2014). Performance of Various Computers Using Standard Linear Equations Software.