

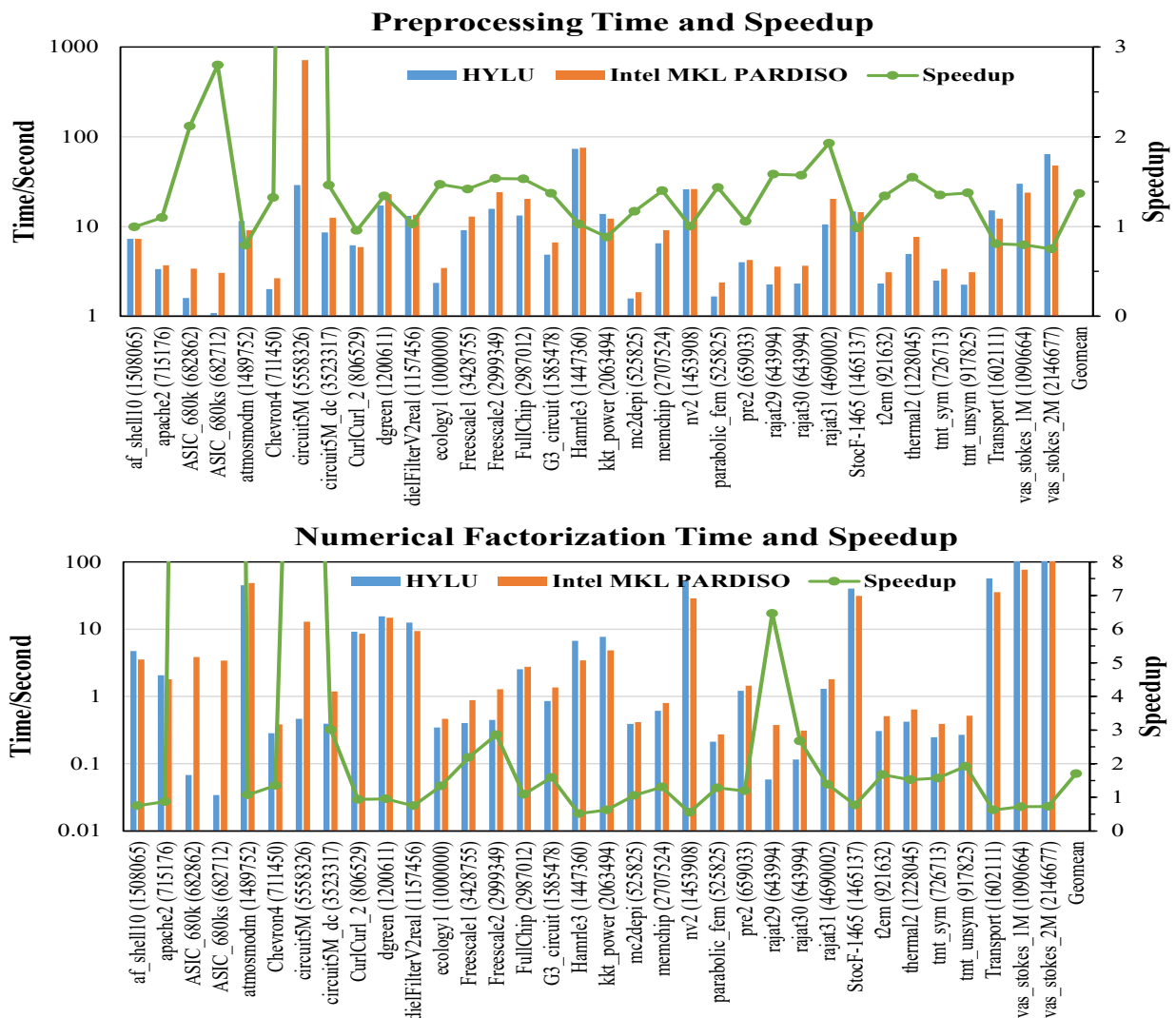
# Test Results of HYLU and Performance Comparisons with Intel MKL PARDISO

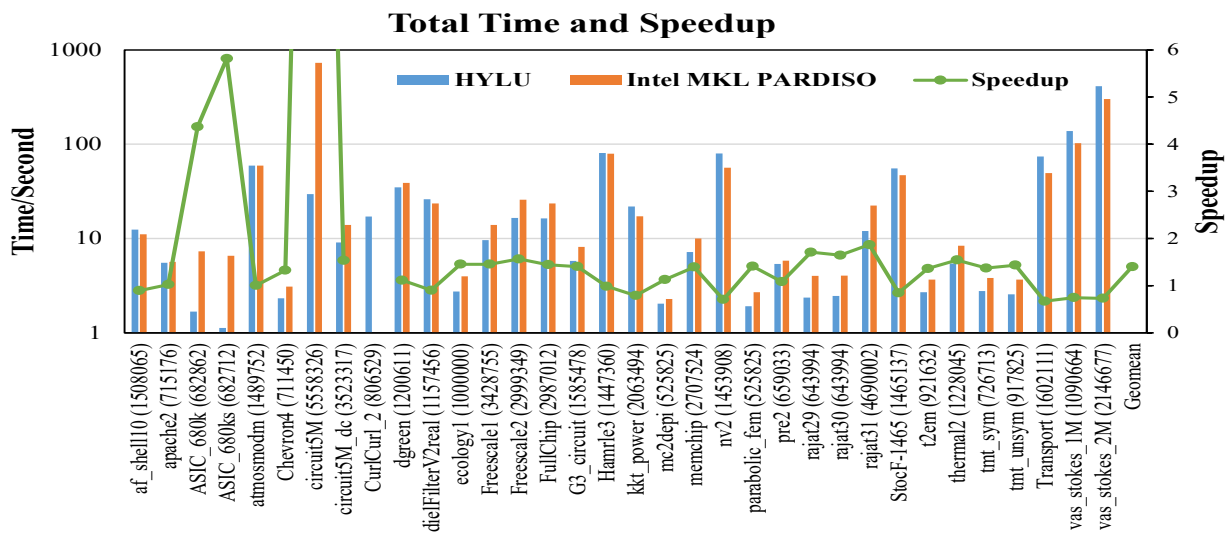
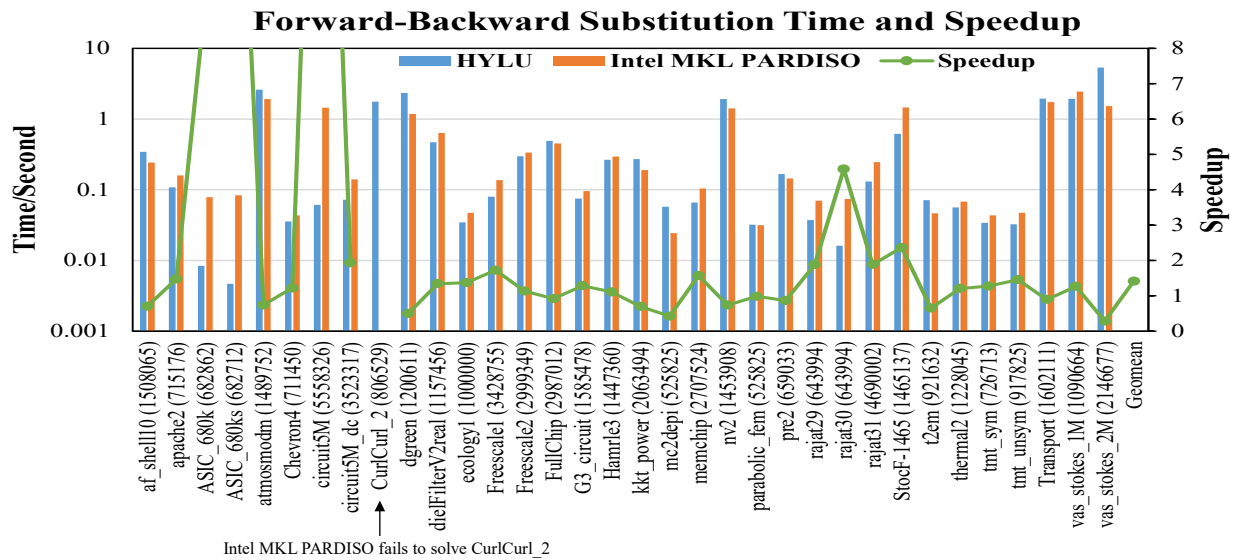
The experiments were carried out on a Linux server. The main hardware and software configurations are listed in the following table. All results presented in this document are wall-time measurements from **16-thread parallel execution**.

|                  |  |
|------------------|--|
| CPU              | Intel Xeon Gold 6130 @ 2.1GHz  |
| Memory           | 256GB  |
| Operating system | Ubuntu 24.04 LTS for tests/CentOS 7.9 for compiling HYLU                             |
| Compiler         | gcc 13.3.0 for compiling test code/gcc 4.8.5 for compiling HYLU                      |
| MKL version      | 2025.2.0.629   |
| Benchmarks       | 34 matrices from SuiteSparse Matrix Collection, dimensions from 525,825 to 5,558,326 |

## 1. One-Time Solve

On geometric mean, HYLU achieves a **1.71X speedup in numerical factorization** compared with Intel MKL PARDISO, while the preprocessing and forward-backward substitution phases are also slightly faster (**1.37X** and **1.41X** speedups, respectively). For the total one-time solve time (preprocessing + numerical factorization + forward-backward substitution), HYLU is **1.40X faster** than Intel MKL PARDISO on geometric mean.





## 2. Repeated Solve

HYLU offers an optimization option for repeated solve of linear systems with an identical sparse pattern in the coefficient matrix. In this case, HYL achieves a **2.21X geometric mean speedup in numerical factorization** over Intel MKL PARDISO, while the forward-backward substitution phase is slightly faster (**1.32X speedup**). When comparing the total time of numerical factorization and forward-backward substitution, HYL is **2.13X** faster than Intel MKL PARDISO on geometric mean. In this scenario, the preprocessing time is on geometric mean 1.84X of that of the one-time solve scenario. However, in the repeated solve scenario, the pre-processing time is less important, as it is executed only once.

