Simulation of Supernova Explosion Accelerated on GPU: Spherically Symmetric Neutrino-Radiation Hydrodynamics

Hideo Matsufuru (High Energy Accelerator Research Organization, KEK)
Kohsuke Sumiyoshi (National Institute of Technology, Numazu College)

Core-collapse supernovae

- Large scale numerical simulation is essential to understand explosion mechanism
  - Hydrodynamics
  - Boltzmann equation for neutrino transport
  - General relativity
  - Equation of state of dense matter, neutrino reactions
- Fully 6D simulations are currently restrictive
  - Dimensionality plays an essential role for explosions
  - Approximations are often used for 2D/3D systematics
  - Acceleration of full 2D/3D simulations are waited

High performance computing

- Massively parallel supercomputer
  - K-computer → Post-K (2021)
  - Intel Xeon Phi
  - Preparation for the next generation SC is underway
- Arithmetic accelerators: GPU, Pezy-SC
  - Heterogeneous architecture
  - *Currently not widely used in spite of large potential*

Spherically symmetric system

- Basis for 2D/3D simulations and observations
- Systematic survey of massive stars is necessary
- 1st principle calculation (Full GR+Hydro.+Boltzmann)
- Testbed for developing GPU code

- Numerical setup
  - GR Lagrangian hydrodynamics + S_N + Implicit scheme

    S. Yamada, ApJ 475 (1997) 720, A&A 344 (1999) 533
  - At every step of time evolution, solve a linear equation
  - BiCGStab algorithms for a block tridiagonal matrix

$$M = \begin{pmatrix} B_1 & C_1 & 0 & & \dots & \\ A_2 & B_2 & C_2 & 0 & & \\ 0 & A_3 & B_3 & C_3 & & \\ \vdots & & \ddots & \ddots & \ddots & 0 \\ & & 0 & A_{n-1} & B_{n-1} & C_{n-1} \\ 0 & \dots & & 0 & A_n & B_n \end{pmatrix}$$

  - Rank of each block matrix: $N_{\rm max} = N_{\rm ang} \cdot N_{E_\nu} \cdot E_\nu + N_{\rm hyd}$
  - Weighted Jacobi preconditioner

    A. Imakura et al., JSIAM Letter 4 (2012) 41

$$x_{k+1} = \omega[-M_D^{-1}(M - M_D)x_k + M_D^{-1}b]$$

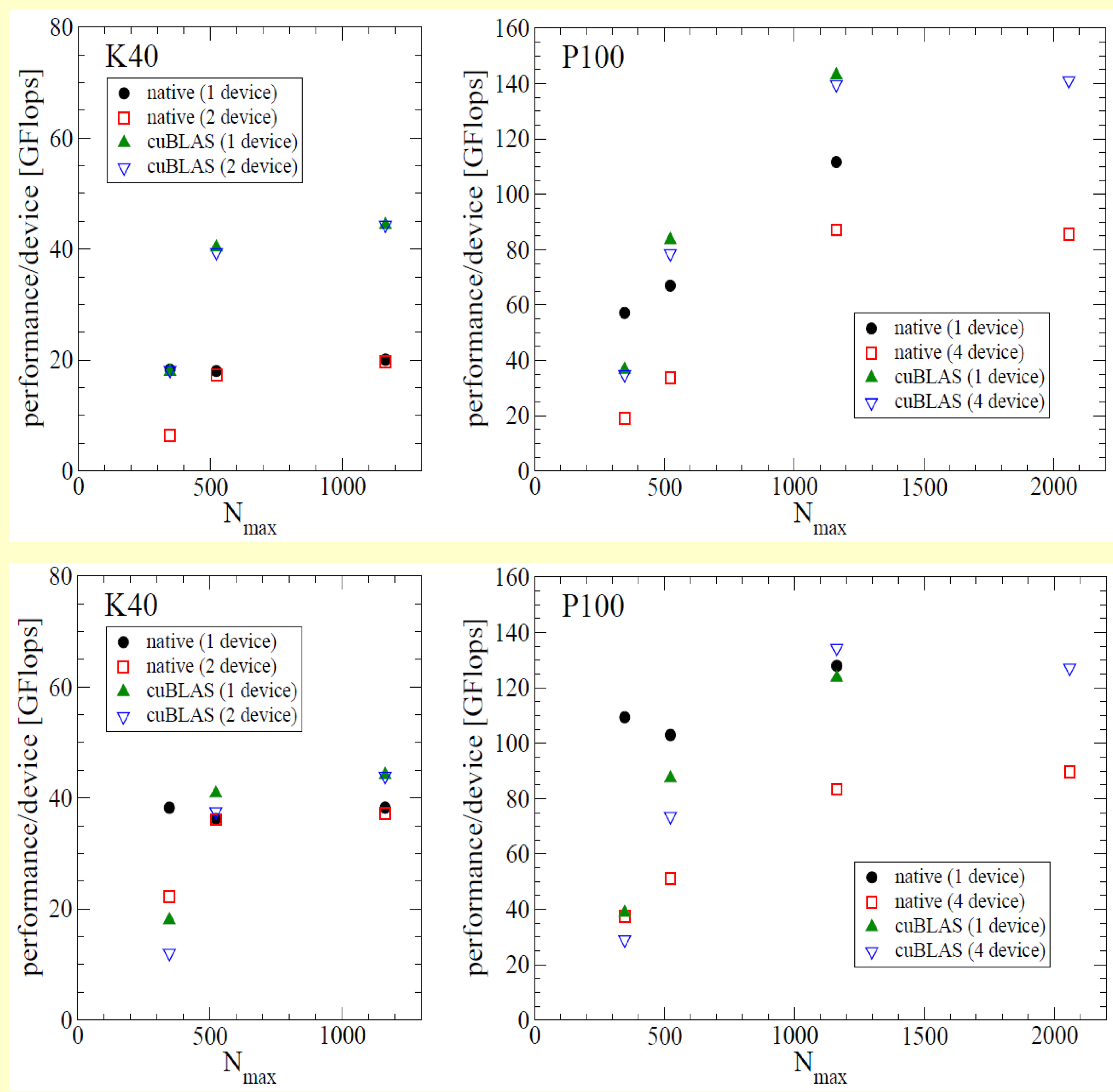
  - This linear equation solver is the first target of offloading

Implementation

- Offloading scheme
  - Device code is implemented with OpenACC
    - Directive-based framework
    - Compiler generates device code
    - Portable, compatible with OpenMP 4.0
  - MPI + multi-GPU
  - Data layout is changed suitably to GPU
- Two implementation of matrix multiplication
  - “native” code
    - Tuned using OpenACC directives
    - Simple assignment of tasks to threads
    - Coalesced access for block matrices
  - cuBLAS code
    - Well-tuned BLAS library by NVIDIA
    - Asynchronous execution: CUDA stream
- Machine setup
  - Intel Xeon + NVIDIA K40 (Kepler) x2
  - IBM Power8 + NVIDIA P100 (Pascal) x 4
    - 1430 and 4700 GFlops/GPU for double precision
  - PGI compiler + CUDA environment
- Performance measurement
  - Inverse of block diagonal matrix (w/o communication)
  - Full/subdiagonal matrices (with communication)
  - Observe $N_{\rm max}$ dependence and scaling with N_r

M_D^{-1}

$M - M_D$



- Results (preliminary)
  - cuBLAS achieves better performance for large $N_{\rm max}$
  - Acceptably high performance for large $N_{\rm max}$ region
  - Simulation with N_r = 1024, $N_{\rm ang} = 12$, $N_{E_\nu} = 24$ becomes practical
- Conclusion
  - GPU largely accelerates the linear solver
  - OpenACC does work as portable and useful framework
  - Acceleration of other parts of simulation is in progress
  - Application to multi-dimensional simulation code

Physics of Core-Collapse Supernovae and Compact Star Formations, Waseda University, Tokyo, Japan, 18-20 March 2018