Adaptive Multistep Predictor for Accelerating Dynamic Implicit Finite-Element Simulations

Kohei Fujita[1,2], Tsuyoshi Ichimura[1,2], Masashi Horikoshi[3], Muneo Hori[1,2], Maddegedara Lalith[1,2]

1. Earthquake Research Institute & Department of Civil Engineering, The University of Tokyo
2. Advanced Institute for Computational Science, RIKEN
3. Software and Solutions Group, Intel K.K.

Introduction

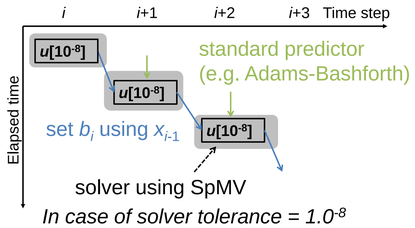
• Data intensive applications have difficulty making use of high arithmetic performance of recent computer hardware
• Dynamic finite-element simulations using iterative solvers based on sparse matrix-vector products provide example of this
• Linear multistep predictors (e.g., Adams-Bashforth method) are widely used to predict initial solution and reduce number of solver iterations
• We aim to improve predictor accuracy using high arithmetic capability of recent computers to further reduce iterations and speed up application

Target problem

Solve dynamic implicit finite-element method using iterative solvers:
input vector *b* is a function of *x* of previous step
Standard algorithm
• Solve sequentially in time; key kernel is memory bandwidth-bound sparse matrix-vector product (SpMV) kernel

Algorithm 1

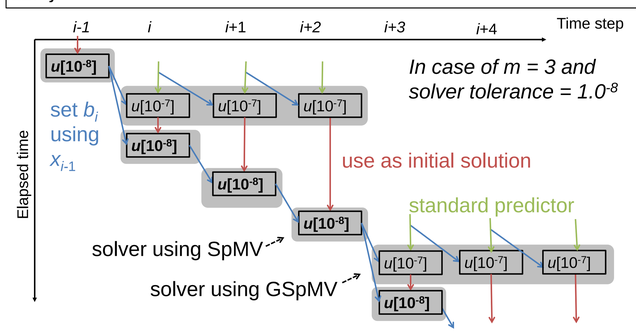
1: set $x_{-1} \leftarrow 0$
2: for($i = 0; i < n; i = i + 1$){
3: guess $\bar{x}_i$ using standard predictor
4: set b_i using x_{i-1}
5: solve $x_i \leftarrow A^{-1}b_i$using initial solution $\bar{x}_i$ **(Computed using SpMV kernel)**
6: }



Multistep predictor algorithm (by Liu et al. [1])
• Predict several time steps simultaneously and use as initial solutions for solvers
• Additional cost required for predictor, but SpMV with multiple right-hand sides (generalized SpMV, or GSpMV) can be used in predictor
• As GSpMV is highly efficient compared with SpMV kernel, reduction in number of iterations in line 9 leads to 30% speedup of whole application

Algorithm 2

1: set $x_{-1} \leftarrow 0$
2: for($i = 0; i < n; i = i + m$){
3: guess $\bar{x}_j$ ($j = i, \dots, i + m - 1$) using standard predictor
4: set b_i using x_{i-1}
5: guess $\bar{b}_j$ using $\bar{x}_{j-1}$ ($j = i + 1, \dots, i + m - 1$)
6: approximately solve $\{\bar{x}_j \leftarrow A^{-1}\bar{b}_j\}$ with initial solution $\bar{x}_j$ ($j = i, \dots, i + m - 1$) **(Computed using GSpMV kernel)**
7: for($j = i; j < i; j = j + 1$){
8: set b_i according to x_{j-1}
9: solve $x_j \leftarrow A^{-1}b_j$using initial solution $\bar{x}_j$ **(Computed using SpMV kernel)**
10: }
11: }

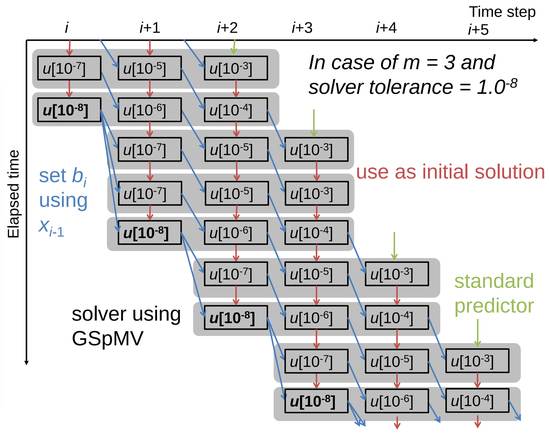


Adaptive Multistep Predictor Algorithm

• ***Adaptively update right hand side vector to improve predictor accuracy***
• In Algorithm 2, predictor accuracy is limited by accuracy of *b* approximated by standard predictor
→ Predictor's approximate solution is different from actual *x*
• Improve predictor accuracy by adaptively updating *b* based on partially-solved *x*
→ Solver in line 8 is interrupted and restarted after updating *b* based on partially-solved *x* (line 6 – 10)
→ Together with predictor accuracy improvement, expect high peak performance since all solvers are based on GSpMV kernels

Algorithm 3

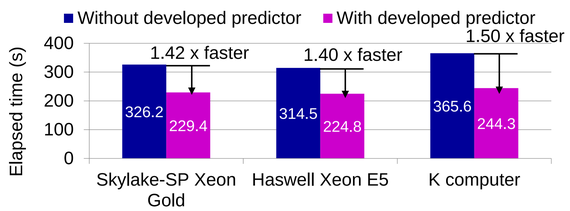
1: set $x_{-1} \leftarrow 0$ and $\bar{x}_i \leftarrow 0$ ($i = 0, \dots, m - 2$)
2: for($i = 0; i < n; i = i + 1$){
3: set b_i using x_{i-1}
4: guess $\bar{x}_{i+m-1}$ using standard predictor
5: $b_j = \bar{b}_j$
6: while ($\frac{|A\bar{x}_i - b_i|}{|b_i|} > \epsilon$) do {
7: guess $\bar{b}_j$ using $\bar{x}_{j-1}$ ($j = i + 1, \dots, i + m - 1$)
8: refine solution $\{\bar{x}_j \leftarrow A^{-1}\bar{b}_j\}$ with initial solution $\bar{x}_j$ ($j = i, \dots, i + m - 1$) **(Computed using GSpMV kernel)**
10: }
11: $x_i = \bar{x}_i$
11: }



Performance Measurement Results

• Compare performance with and without developed predictor (Algorithm 1 and 3 [*m* = 4]) for 25 time steps of an earthquake ground motion problem discretized with second-order tetrahedral elements
• Both solvers implemented based on SC15 Gordon Bell Prize Finalist solver [2]. Highly-efficient implementation, optimized to reflect algorithm changes

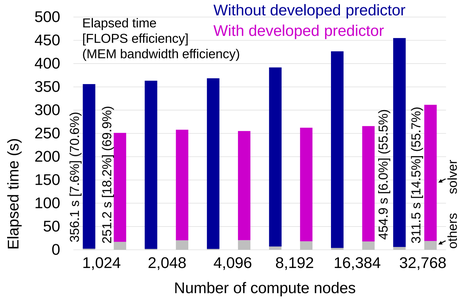
Performance of algorithm



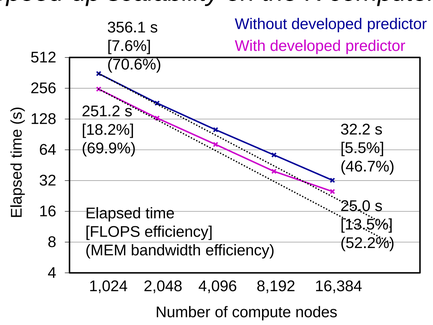
	20 cores x 2 sockets x 2 nodes @ 2.4 GHz	12 cores x 2 sockets x 4 nodes @ 2.6 GHz	8 cores x 1 socket x 8 nodes @ 2.0 GHz
Peak DP FLOPS	3072 GFLOPS	3993.6 GFLOPS	1024 GFLOPS
Peak mem. bandwidth	512 GB/s	544 GB/s	512 GB/s

• Number of iterations: 9,068 → 3,541 (factor of 2.56)
• Elapsed time reduced by 40%-50% on Skylake-SP Xeon Gold, Haswell Xeon E5, and K computer
• High performance for unstructured finite-element method: 14.5% of DP peak attained for whole solver on 32,768 K computer nodes
• High scalability attained up to 32,768 K computer nodes: 80.6% size-up efficiency
• Faster time-to-solution for all speed-up scalability problems on K computer

Size-up scalability on the K computer

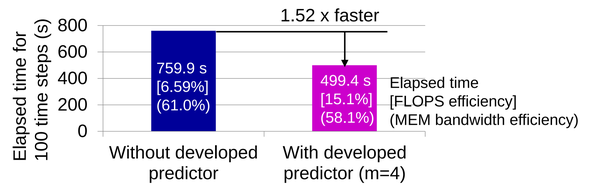


Speed-up scalability on the K computer



Application to a Practical Problem

• 10.25 km x 9.25 km x 280 m area
• 3 soil layers
• 4 m second-order tetrahedral elements
• 931,570,385 nodes, 686,595,280 elements, 2,794,711,155 degrees of freedom



• Number of iterations reduced by factor of 2.62 and 52% speedup for practical problem of 10.25 km x 9.25 km x 280 m area of central Tokyo discretized using 4 m elements computed using 2,304 K computer nodes

Summary and Future Prospects

• Developed adaptive multistep predictor for accelerating dynamic implicit finite-element simulations
• Achieved speedup for wide range of CPU-based environments by reducing number of iterative solver iteration using highly efficient GSpMV kernel
• Adaptive multistep predictor expected to be useful for wide range of dynamic implicit problems, as it is applicable to more sophisticated iterative solvers (i.e., multi-grid iterative solvers)

Acknowledgements

Our results were obtained using the K computer at the RIKEN Advanced Institute for Computational Science (proposal numbers: hp160221, hp160160, hp160157, and hp170249). We acknowledge the support provided by the Japan Society for the Promotion of Science (15K18110, 26249066, 25220908, and 17K14719). We thank the Operations and Computer Technologies Division of RIKEN AICS and the High Performance Computing Infrastructure helpdesk for their support in the use of the K computer.

References

[1] X. Liu, E. Chow, K. Vaidyanathan, and M. Smelyanskiy. "Improving the Performance of Dynamical Simulations Via Multiple Right-Hand Sides," IEEE 26th International Parallel and Distributed Processing Symposium, 2012.
[2] T. Ichimura, K. Fujita, P.E.B. Quinay, L. Maddegedara, M. Hori, S. Tanaka, Y. Shizawa, H. Kobayashi, and K. Minami. "Implicit nonlinear wave simulation with 1.08T DOF and 0.270T unstructured finite elements to enhance comprehensive earthquake simulation," Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis, (SC'15), 2015.