Advanced Energy Systems and Intelligent Geoscience (AESIG)
https://www.mospbs.com/journal/aesig

• Review •

Research Progress on CPU/GPU Heterogeneous Parallelism and Efficient Solvers in Large-Scale Complex Reservoir Simulation

Kang Wang[1], **Xiang Rao**[1,2,3,*]

[1] School of Petroleum Engineering, Yangtze University, Wuhan 430100, China.

[2] State Key Laboratory of Low Carbon Catalysis and Carbon Dioxide Utilization (Yangtze University), Wuhan 430100, China.

[3] Western Research Institute, Yangtze University, Karamay 834000, China.

[*]**Corresponding Author:** Xiang Rao **Email:** raoxiang0103@163.com

Received: 07 January 2026 **Accepted:** 12 January 2026

Abstract: This paper aims to systematically review the latest evolutionary routes of High-Performance Computing (HPC) in the field of reservoir numerical simulation. First, starting from the underlying hardware architecture, the article details the fundamental differences in data transmission mechanisms, programming complexity, and acceleration performance between the CPU/GPU hybrid offload mode and the full GPU native architecture. Second, it deeply analyzes key algorithmic breakthroughs for fractured reservoirs, focusing on the parallel assembly strategy for Non-Neighbor Connections (NNC) in the Embedded Discrete Fracture Model (EDFM) on GPUs, as well as multi-color DILU and CPR-AMG two-stage preconditioning technologies adapted for GPU many-core architectures. Third, based on industrial benchmark data from mainstream simulators such as tNavigator, Stone Ridge Echelon, and the open-source OPM Flow, the acceleration effectiveness and scalability of heterogeneous parallelism are quantitatively evaluated across models of varying scales. Finally, the paper critically points out deep-seated contradictions in current technologies regarding calculation accuracy on non-K-orthogonal grids, the VRAM capacity "ceiling" effect, and cross-platform portability, while looking forward to the convergent development trends of mixed-precision computing and physics-constrained AI preconditioning technologies.

Keywords: Large-scale reservoir simulation; GPU acceleration; Heterogeneous parallelism; Linear solvers; Embedded Discrete Fracture Model (EDFM); Non-Neighbor Connections (NNC)

1 Introduction

As global oil and gas exploration expands into deep formations, deep water, and unconventional domains, the precision requirements for geological models have escalated from millions of grid cells to billions. Simultaneously, the multi-scale fracture networks developed in complex reservoirs, such as shale gas and vuggy carbonate rocks, introduce extreme heterogeneity and anisotropy.

DOI: https://doi.org/10.58244/aesig.263495

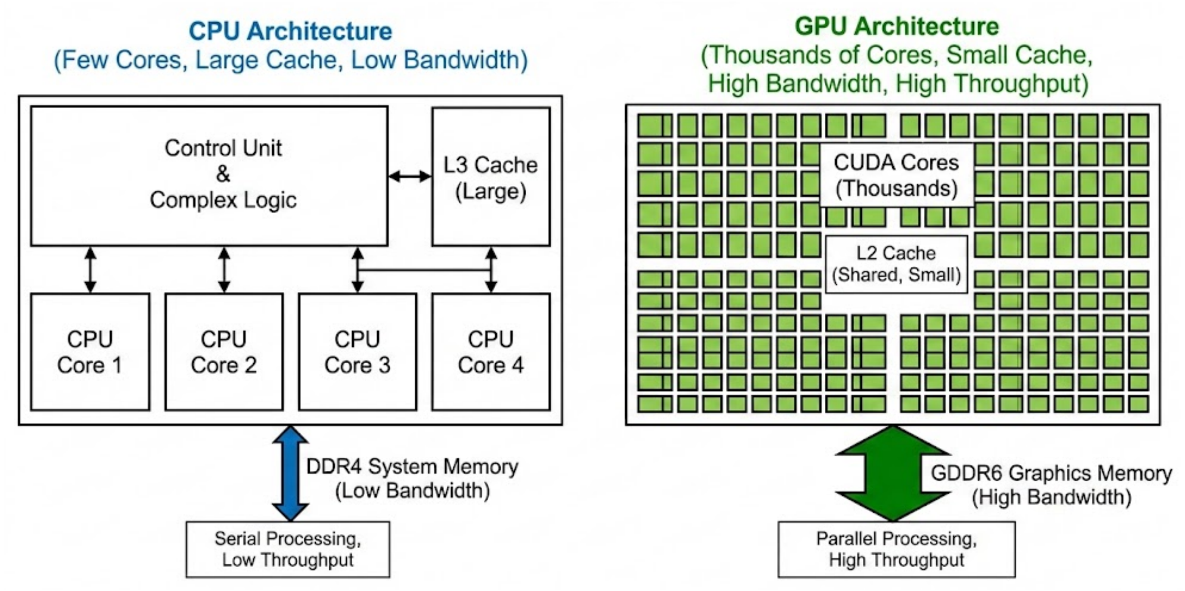
CPU Architecture
(Few Cores, Large Cache, Low Bandwidth)
Control Unit & Complex Logic
L3 Cache (Large)
CPU Core 1
CPU Core 2
CPU Core 3
CPU Core 4
DDR4 System Memory (Low Bandwidth)
Serial Processing, Low Throughput
GPU Architecture
(Thousands of Cores, Small Cache, High Bandwidth, High Throughput)
CUDA Cores (Thousands)
L2 Cache (Shared, Small)
GDDR6 Graphics Memory (High Bandwidth)
Parallel Processing, High Throughput

Fig.1 Comparison of CPU vs. GPU Architectural Design Philosophies and Bandwidth Performance Differences.

This leads to severe deterioration in the condition number of the Jacobian matrix in linear systems, posing immense computational challenges to traditional numerical simulation methods. Traditional CPU cluster parallel technologies based on the Message Passing Interface (MPI) are constrained by the "Memory Wall" effect in which memory bandwidth growth lags behind the increase in computational cores, as well as by the non-linear accumulation of inter-node communication latency. Consequently, meeting the industry's urgent demand for refined simulation within reasonable time and energy costs has become difficult [1].

Numerical simulation acts as the "industrial brain" of reservoir engineering decision-making, playing an irreplaceable role in remaining oil distribution prediction, development plan optimization, and Enhanced Oil Recovery (EOR) assessment. In recent years, with the popularization of "broadband and wide-azimuth" seismic data processing technologies and the application of multi-point geostatistical modeling methods, the spatial resolution of geological models has significantly improved. Grid counts for single-well models have generally reached the million level, while full-field models have surpassed the billion scale [2, 3]. Particularly for complex reservoirs such as shale gas hydraulic fracture networks and carbonate vuggy systems, their multi-scale flow characteristics (ranging from millimeter-scale fractures to kilometer-scale faults) require extremely high grid density. This causes the computational load for simulations involving complex physical processes like multi-component fluids and thermal recovery to grow exponentially.

However, there is a significant "scissors difference" (gap) between the supply of computing power and demand. Over the past thirty years, the industry has primarily relied on CPU-based Domain Decomposition techniques to address scaling challenges. Although the core count of modern CPUs has continuously increased (from single-core to 64-core or more), their architectural design has encountered an insurmountable "Memory Wall" bottleneck , in which the annual growth rate of memory bandwidth (approximately 10%)

is far lower than the growth rate of computational core performance (approximately 50%).

As shown in Figure 1, the most time-consuming steps in reservoir simulation are Sparse Matrix-Vector multiplication (SpMV) and linear system solving, which are typical Memory-bound tasks. A vast number of transistors in CPUs are dedicated to branch prediction and out-of-order execution. However, when performing linear algebra operations on massive datasets, CPU calculation cores often sit idle waiting for data transmission from memory to cache, resulting in low actual parallel efficiency. In contrast, Graphics Processing Units (GPUs), with their high-throughput parallel architecture designed for graphics rendering and High Bandwidth Memory (HBM, typically 10-20 times that of contemporary CPUs), provide a new physical foundation for breaking this bottleneck [4]. From early academic code exploration to the mature application of commercial software like tNavigator and Stone Ridge Echelon today, GPU acceleration has evolved from an auxiliary method into a major driver of numerical simulation, enabling simulations that more closely reflect real physical processes.

2 Evolution of Parallel Architectures

To address the computational demands of large-scale models, the utilization of hardware resources has evolved through three stages: from homogeneous computing to heterogeneous collaboration. This chapter details the technical characteristics and applicable scenarios of these three mainstream architectures.

2.1 Traditional CPU Massive Parallelism (CPU-based MPI)

This is currently the most mature and widely applied solution, adopted by classic simulators like Eclipse Parallel and CMG GEM. Its core idea involves using graph partitioning libraries such as METIS or ParMETIS to slice the massive geological grid into multiple sub-domains with relatively balanced loads, assigning them to different CPU cores. Cores utilize the MPI (Message Passing Interface) protocol to transmit boundary “Ghost Cell” data at each time step and linear iteration step [14].

However, this architecture faces severe scalability bottlenecks. According to Amdahl’s Law, the system’s speedup is limited by the serial portion and communication overhead. As the number of computing cores increases, inter-node communication overhead grows non-linearly at $O(N^2)$ or $O(N\log N)$. Research indicates that when the number of grid cells per sub-domain falls below 10,000, the time consumed by boundary data exchange exceeds the time for internal domain calculation, leading to a “plateau” or even a regression in Strong Scaling performance [5]. Furthermore, the high energy consumption (power and cooling) and floor space costs required to build and maintain large-scale CPU clusters have prompted the industry to seek new computing paradigms with higher energy efficiency ratios.

2.2 CPU/GPU Hybrid Offload Mode (Hybrid Offload)

Considering the immense engineering effort and risk involved in completely rewriting simulator code, organizations like SLB Intersect and TotalEnergies have adopted a compromise hybrid offload strategy [6, 7]. In this mode, the simulator retains CPU code validated over decades to handle logically complex well control management (e.g., multi-well production network coupling), PVT phase equilibrium (Flash) calculations, and adaptive time-step scheduling. It offloads only the Linear Solver, which accounts for 70%-90%

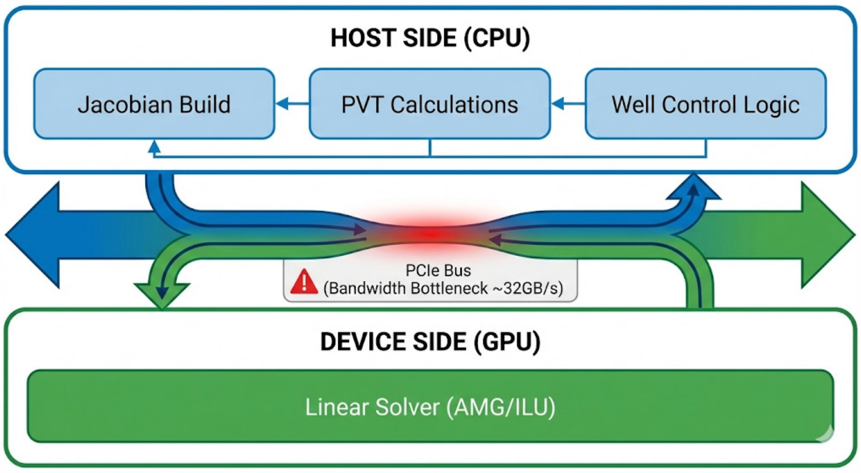

Fig.2 Data Flow and PCIe Bottleneck in CPU/GPU Hybrid Heterogeneous Computing Architecture

of the total runtime, to the GPU for accelerated calculation.

As shown in Figure 2, while this mode protects legacy code assets, its fatal weakness lies in the limitation of the PCIe bus bandwidth. During implicit solving, massive Jacobian matrices and residual vectors must be transmitted between the Host and Device at every Newton iteration step. For small to medium-scale models (e.g., fewer than 1 million grids), this repetitive data transfer time often cancels out the speed dividend brought by GPU calculation, a phenomenon known as "acceleration negated by transmission". Therefore, the hybrid mode usually only demonstrates significant acceleration advantages when processing large-scale models with over ten million grid cells [7].

2.3 Native GPU Mode (Native GPU)

To pursue extreme computational performance, new-generation simulators represented by Stone Ridge (Echelon) and tNavigator have chosen a "clean slate" technical route [3, 8]. In this full GPU architecture, the simulator's core computational kernels are completely rewritten. Complex Jacobian Assembly, fluid high-pressure property updates, porosity/permeability parameter calculations, and the entire linear solving process run resident in video memory (VRAM). The CPU is responsible only for extremely simple logic control and I/O management.

This architecture completely eliminates data transmission overhead between CPU and GPU, allowing data to flow continuously within ultra-high bandwidth VRAM, realizing true "Zero-copy" computing. Esler et al. [3] reported that on the SPE10 extended model (hundreds of millions of grid cells), the full GPU architecture achieved a speedup of over 50 times compared to traditional CPU clusters. However, the cost is an extremely high development threshold (requiring millions of lines of code to be rewritten using heterogeneous programming languages like CUDA) and strong dependence on hardware VRAM capacity. Processing ultra-large-scale models necessitates reliance on expensive multi-card interconnection technologies like NVLink to build a unified memory pool.

3 Core Challenges: Efficient Solvers Adaptation for GPU

Regardless of the hardware architecture, solving ultra-large-scale sparse linear systems Ax=b remains the most time-consuming core pain point in simulators. Traditional CPU solving algorithms (such as strongly recursive ILU) cannot be directly transplanted to GPUs possessing thousands of cores due to strong data dependencies; they must undergo thorough algorithmic reconstruction to adapt to the SIMT (Single Instruction, Multiple Threads) architecture.

3.1 Fine-Grained Parallel Preprocessing Technology: Multi-Color DILU

The quality of the preconditioner determines the convergence speed of the iterative method, while its degree of parallelism determines computational throughput. Traditional ILU(0) factorization performs excellently on CPUs, but its Forward and Backward Substitution processes have strong serial recursiveness (calculation of the next row depends on the result of the previous row), making them essentially impossible to parallelize on GPUs, resulting in extremely low thread utilization.

Andersen et al. [2] reported that Graph Coloring technology can be used to reorder the matrix. By algorithmically marking non-adjacent (i.e., independent) grid cells with the same color, all grid cells of the same color can be processed completely in parallel on the GPU, requiring synchronization only between different colors. Experimental results show that in the OPM Flow simulator, the optimized multi-color GPU-DILU solver is 10-20 times faster than the CPU version of ILU(0). Although multi-color ordering mathematically slightly reduces the quality of preprocessing (breaking transmission information under natural ordering, leading to more linear iterations), the total solution time is significantly shortened due to the massive increase in single iteration speed represents a typical "trading computation for time" strategy.

3.2 CPR-AMG Two-Stage Solver Adapting to Strong Heterogeneity

For fractured reservoirs, simple single-stage preprocessing often fails to effectively eliminate errors due to the flow capacity difference between fractures and matrix potentially reaching 10^4 or higher, leading to non-convergence of linear iterations. Mohajeri et al. [9] proposed a full GPU version of the CPR-AMG (Constrained Pressure Residual-Algebraic Multigrid) strategy.

As shown in Figure 3, this strategy divides the solving process into two stages:

Global Pressure Solution (Stage 1): Uses Algebraic Multigrid (AMG) to effectively eliminate low-frequency pressure errors on the coarse grid, resolving long-distance pressure transmission issues. To adapt to GPU architecture, Andersen et al. [2] adopted Aggregation-based coarsening algorithms instead of the classic Ruge-Stuben algorithm, as the former has simpler logic, is easier to parallelize on GPUs, and effectively reduces Setup phase time.

Local Saturation Solution (Stage 2): Uses GPU-accelerated ILU or Gauss-Seidel to solve

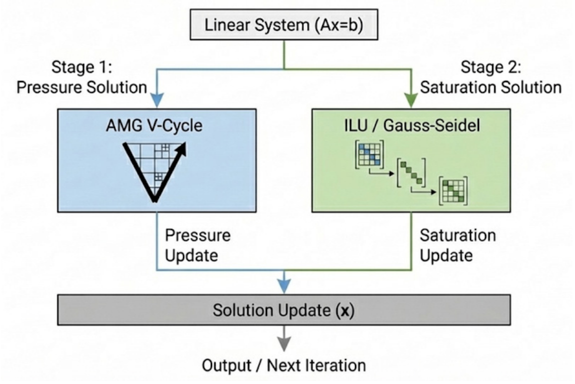

Fig.3 Schematic of the CPR-AMG Two-Stage Preconditioning Strategy for Coupled Pressure-Saturation Systems (Mohajeri et al. [9])

saturation equations, handling high-frequency local errors.

This combined strategy demonstrated strong robustness when processing complex fracture benchmark models provided by Sharif University, solving convergence issues that pure GPU-ILU could not handle. It is currently the standard paradigm in the industry for solving strong heterogeneity problems.

4 Special Techniques for Fractured Reservoirs

Simulations of fractured reservoirs are not only large in scale but also present special challenges for GPU parallelism due to the complexity of their geometric structures, particularly regarding memory access optimization and load balancing.

4.1 Parallel Assembly of Embedded Discrete Fracture Model (EDFM)

The EDFM "embeds" fracture patches onto background grids, avoiding the difficulties of unstructured meshing, but introduces massive Non-Neighbor Connections (NNC) [18-21]. Establishing transmissibility relationships between grids that are physically non-adjacent but mathematically connected causes the sparse structure of the Jacobian matrix to become extremely irregular, with significantly widened bandwidth.

The existence of NNC causes two severe problems when assembling the Jacobian matrix on a GPU: first, Race Conditions, where multiple grids (matrix and fracture) may simultaneously accumulate flux into the same matrix element; second, discontinuous VRAM access, where the random distribution of NNC data severely undermines the GPU's Coalesced Access mechanism. Chen et al. [10] and Zhang et al. [11] proposed a parallel assembly algorithm based on Reordering. By preprocessing the NNC table, non-conflicting connections are grouped together, and compressed storage formats optimized for GPUs (such as Sliced ELLPACK) are adopted. This approach not only avoids inefficient Atomic Operations but also significantly improves VRAM read efficiency.

4.2 Dynamic Load Balancing

In fracture-matrix systems, the fracture region exhibits extremely fast flow rates and strong nonlinearity, often requiring more Newton iterations and smaller time steps, while matrix region calculations are relatively simple. Simply decomposing domains based on spatial geometry leads to overload on GPU cores responsible for fracture regions, while cores responsible for the matrix remain idle, resulting in a distinct "wooden barrel effect". Lian et al. [12], in simulation practices for the Sinopec Tahe Oilfield, adopted a dynamic domain decomposition algorithm based on Graph Weight. This algorithm dynamically assigns higher weights to fracture grids based on calculation time from the previous time step, adjusting the number of grids assigned to each GPU node. Field tests showed that this strategy controlled the calculation time difference between nodes to within 5%, increasing the overall speedup by 25% and effectively solving the computational load imbalance caused by heterogeneous media.

5 Case Studies and Performance Evaluation

Based on measured data from literature, this chapter quantitatively evaluates the real-world implementation effects of GPU technology from an industrial application perspective, comparing performance differences with traditional CPU solutions.

5.1 tNavigator vs. Eclipse (Industrial Black Oil Model)

According to detailed comparative test reports by Luo et al. [8] and Bogachev et al. [13], in a test case based on a domestic marine sandstone reservoir (330,000 grids, containing 14 years of production history and dozens of multi-lateral horizontal wells), a single-node GPU workstation (equipped with NVIDIA Tesla P100) was compared with a 32-core high-performance CPU cluster.

Test results indicate that the fullprocess calculation time of tNavigator running on GPU was only about onethird of that of Eclipse on CPU. In additional experiments with a largescale model containing 18 million grids, simulations completed within hours using four Tesla P100 cards, and the acceleration exhibited a nearly linear growth trend as the grid size increased. These findings highlight the highthroughput advantage of GPUs in handling largescale reservoir simulations and suggest that a desktoplevel GPU workstation can effectively replace traditional rackmounted CPU clusters, thereby significantly reducing hardware procurement and operational costs.

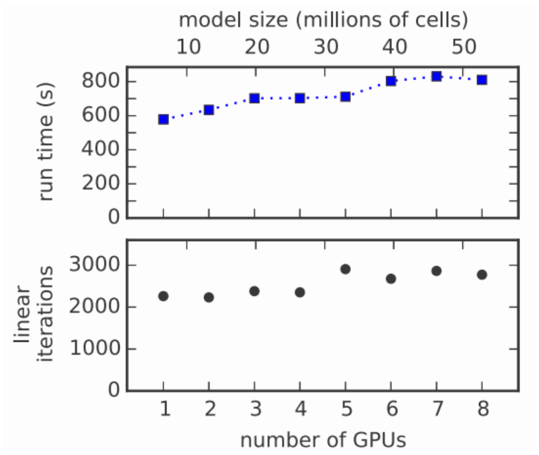

Fig.4 Weak Scaling Performance of the Stone Ridge Echelon Model on 8 GPUs (Esler et al. [3])

5.2 Stone Ridge Echelon (Billion-Grid Limit Test)

Esler et al. [3] performed an array replication extension on the standard SPE10 model to build an ultra-large-scale test model reaching hundreds of millions of grids to verify the limit capabilities of full GPU architecture.

The test results show near-ideal Weak Scaling on 8 GPUs, meaning as computational resources and grid scale increase proportionally, the calculation time per unit grid remains constant. This powerfully proves the absolute dominance of GPU high-bandwidth VRAM in large-scale implicit solving, indicating that full GPU architecture is capable of meeting future simulation demands for billion-grid models.

6 Challenges & Limitations

6.1 The VRAM Capacity “Ceiling” Effect

VRAM capacity is the primary factor constraining GPU simulation scale. Current top-tier computing cards (e.g., NVIDIA H100) have approximately 80 GB of VRAM, whereas a billion-grid compositional model with 10 components may require terabytes of memory space for its Jacobian matrix. Although multi-card interconnection is possible, communication latency between GPUs (via NVLink or PCIe) remains a bottleneck when processing cross-node Long Fractures. Strong scalability for large-scale implicit solving in multi-GPU environments still awaits breakthroughs, especially when the model scale exceeds the sum of single-node VRAM, necessitating cross-node MPI communication which significantly reduces acceleration efficiency.

6.2 Accuracy Risks under Non-K-Orthogonal Grids

To pursue data alignment and high

throughput, current commercial GPU simulators typically use Two-Point Flux Approximation (TPFA). However, in Non-K-Orthogonal grids common in actual geological models (e.g., severely distorted corner-point grids, PEBI grids), TPFA ignores the cross-terms of the permeability tensor, leading to significant Grid Orientation Effects. Adopting Multi-Point Flux Approximation (MPFA) or Mimetic Finite Difference (MFD) method to correct errors requires accessing data from non-neighboring grids (extended 9-point or 27-point stencils), which severely disrupts the GPU's coalesced memory access patterns, causing performance to drop precipitously [14, 22, 23]. Achieving unity between "high-order accuracy formats" and "memory access continuity" on GPUs remains an unsolved problem.

6.3 Lack of Heterogeneous Algorithm Frameworks and Poor Portability

Currently, 90% of industrial GPU code is written based on NVIDIA CUDA, creating a highly closed ecosystem. With geopolitical factors, the domestic oil and gas industry faces risks of hardware restrictions. Existing code is difficult to directly transplant to domestic GPUs (e.g., Hygon DCU, Huawei Ascend), and there is a lack of general heterogeneous programming standards similar to MPI. Furthermore, existing CPU/GPU collaborative modes lack dynamic load balancing standards, making it difficult to flexibly cope with drastic fluctuations in computational load during production (e.g., after water breakthrough).

7 Future Trends

7.1 Mixed Precision Computing

Utilizing the FP16 (half-precision) or FP8 capabilities of NVIDIA Tensor Cores to accelerate the preprocessing stage and residual calculation of linear solvers, while using FP32/FP64 to ensure solution updates and physical conservation. Preliminary attempts by Yang et al. [15] indicate that this strategy is expected to increase speed by another 2 times without losing physical accuracy, while significantly reducing VRAM usage. This is a crucial direction for breaking the "Memory Wall".

7.2 Deep Fusion of AI and Physics Simulation (AI-Physics)

Yang et al. [15] demonstrates an "AI Geology-Engineering Workflow". The future trend is not to completely replace numerical simulation with AI, but to build an efficient "AI Coarse Screening + GPU Precise Calculation" workflow: using Graph Neural Networks (GNN) to predict the initial value of the pressure field and serving it as the Initial Guess for the linear solver. This can reduce linear iterations by 30%-50%, thereby achieving acceleration within a strict physical solver framework, guaranteeing physical conservation while leveraging AI inference speeds.

7.3 Meshfree Reservoir Simulation Methods

In recent years, meshfree reservoir numerical simulation methods have emerged, such as generalized finite difference and extended finite volume. These methods discretize the reservoir domain using point clouds, which are subject to far weaker topological constraints than traditional grids, thereby enabling a more natural treatment of complex reservoir geometries (e.g., fractures, karst caves, faults, and irregular boundaries). In meshfree methods, local differential operators must generally be obtained node by node, and the computation of differential operators at different nodes is mutually independent, exhibiting inherent parallelism. GPU acceleration is therefore crucial for improving the computational efficiency of meshfree simulation methods and shows signifi-

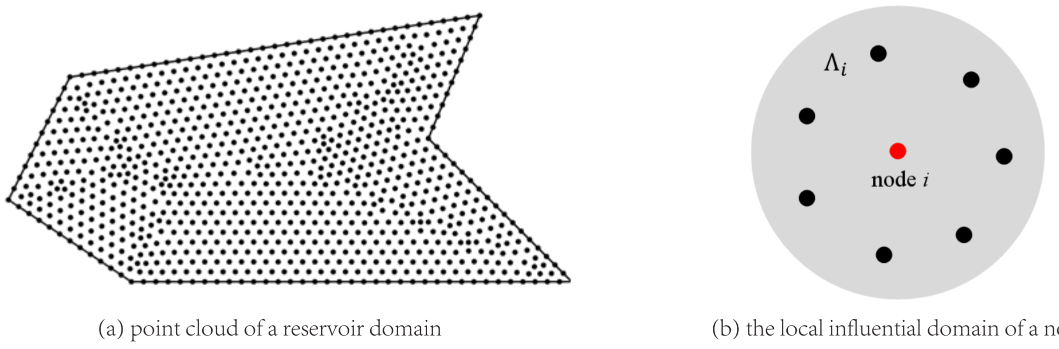
Λ_i
node i
(a) point cloud of a reservoir domain
(b) the local influential domain of a node

Fig.5 Point Cloud Discretization and Local Influence Domain in Meshfree Reservoir Simulation

cant application potential [24-27].

8 Conclusion

In summary, this study systematically reviewed the current progress and challenges of GPUaccelerated reservoir simulation. The conclusion highlights three key aspects: architecture choice, core technology development, and future outlook. These perspectives collectively demonstrate both the opportunities and limitations of applying manycore parallel computing to complex reservoir models.

(1) For large-scale models exceeding ten million grids, full GPU architecture is the optimal performance solution, capable of achieving 10-50x acceleration; for legacy models with complex logic and massive legacy code, the hybrid offload mode is a realistic compromise.

(2 Simple transplantation of CPU algorithms leads to a dead end. Fine-grained parallel algorithms designed specifically for GPU many-core architectures (such as multi-color DILU, aggregation AMG) and NNC reordering strategies for EDFM must be developed to maximize VRAM bandwidth utilization.

(3) The current primary bottleneck has shifted from “computation speed” to “VRAM capacity” and “physical accuracy”. Future breakthroughs lie in mixed-precision computing, physics-constrained AI preprocessing, and deep adaptation to domestic heterogeneous hardware to achieve autonomous and controllable high-performance reservoir simulation technology.

Acknowledgement

None.

Funding Statement

None.

Author Contributions

The author confirms sole responsibility for the following: study conception and design, datacollection, analysis and interpretation of results, and manuscript preparation.

Availability of Data and Materials

None.

Conflicts of Interest

The authors declare that they have no conflicts of interest to report regarding the present study.

Reference

[1] Meng, X., He, X., Hu, C., & Lu, X. (2025). A review of parallel computing for large-scale reservoir numerical simulation. Archives of Computational Methods in Engineering.

[2] Andersen, T. M., Torben, J., Lye, K. O., & Lie, K. A. (2025, March). A comparison of DILU and ILU(0) as GPU-accelerated preconditioners. SPE Reservoir Simulation Conference. Society of Petroleum Engineers.

[3] Esler, K., Mukundakrishnan, K., Natoli, V., Shumway, J., Zhang, Y., & Gilman, J. (2014, September). Realizing the potential of GPUs for reservoir simulation. 14th European Conference on the Mathematics of Oil Recovery (ECMOR XIV). European Association of Geoscientists & Engineers.

[4] Liu, H., Yu, S., Chen, Z., Hsieh, B., & Shao, L. (2012, April). Parallel preconditioners for reservoir simulation on GPU. SPE Latin America and Caribbean Petroleum Engineering Conference. Society of Petroleum Engineers.

[5] Liu, H., Wang, K., Chen, Z., Jordan, K. E., Luo, J., & Deng, H. (2015, October). A parallel framework for reservoir simulators on distributed-memory supercomputers. SPE/IATMI Asia Pacific Oil & Gas Conference and Exhibition. Society of Petroleum Engineers.

[6] Appleyard, J. R., Appleyard, J. D., Wakefield, M. A., & Desitter, A. L. (2011, February). Accelerating reservoir simulators using GPU technology. SPE Reservoir Simulation Symposium. Society of Petroleum Engineers.

[7] Cao, H., Zaydullin, R., Liao, T., Gohaud, N., Obi, E., & Darche, G. (2021, October). Adding GPU acceleration to an industrial CPU-based simulator: Development strategy and results. SPE Reservoir Simulation Conference. Society of Petroleum Engineers.

[8] Luo, D., Qiao, C., Gu, Y., & Zhang, J. (2020). tNavigator: A fine reservoir simulator based on modern CPU and GPU computing platforms assisting efficient development of large oil and gas fields. Computer Knowledge and Technology, 16(18), 205–206.

[9] Mohajeri, S., Eslahi, R., Bakhtiari, M., Alizadeh, A., Zeinali, M., Madani, M., Rajabi, H., Sharifi, E., Mortezazadeh, E., & Mahdavifar, Y. (2020, November). An adaptive CPR-AMG based linear solver for simulating geometrically complicated and fractured reservoirs. Abu Dhabi International Petroleum Exhibition & Conference. Society of Petroleum Engineers.

[10] Chen, Y., Zhang, D., Cui, S., & Li, J. (2020). Research on heterogeneous parallel algorithms for numerical simulation of dual-porosity reservoirs. Computer Engineering and Science, 42(10), 1880–1886.

[11] Zhang, K., Chen, Z., & Liu, H. (2022). GPU-accelerated EDFM for simulation of fractured reservoirs. Journal of Petroleum Science and Engineering, 208, 109358.

[12] Lian, P., Ji, B., Duan, T., & Zhao, H. (2020). CPU and GPU hybrid parallel numerical simulation for large complex reservoirs. China Sciencepaper, 15(5), 537–541.

[13] Bogachev, K., Milyutin, S., Telishev, A., Nazarov, V., Shelkov, V., & Eydinov, D. (2018). High-performance reservoir simulations on modern CPU-GPU computational platforms. Rock Flow Dynamics Technical Report.

[14] Wu, S., Li, J., Zhang, Y., & Chen, Z. (2021). Challenges in parallel reservoir simulation. SPE Journal, 26(5), 2345–2356.

[15] Yang, X., Zhang, D., Liu, D., Cao, L., Qu, C., Liu, C., & Yu, W. (2025, March). AI-driven geology-engineering workflow for hydraulic fracture evaluation with GPU acceleration. SPE Reservoir Simulation Conference. Society of Petroleum Engineers.

[16] Lin, M., Tayir, Zou, J., Jing, S., & Guan, Y. (2013). Application prospects of GPU computing in oil and gas exploration. Computer Systems & Applications, 22(3), 6–10.

[17] Bernaschi, M., Celestini, A., D'Ambra, P., & Vella, F. (2023). Multi-GPU aggregation-based AMG pre-

conditioner for iterative linear solvers. arXiv .

[18] Rao, X., He, X., Du, K., Kwak, H., Yousef, A., & Hoteit, H. (2024). A novel projection-based embedded discrete fracture model (pEDFM) for anisotropic two-phase flow simulation using hybrid of two-point flux approximation and mimetic finite difference (TPFA-MFD) methods. Journal of Computational Physics, 499, 112736.

[19] Rao, X., Guo, S., He, X., Kwak, H., Yousef, A., & Hoteit, H. (2025). A first streamline-based simulation method within the projection-based embedded discrete fracture model (pEDFM). Computers and Geotechnics, 185, 107357.

[20] Rao, X., He, X., Xu, Y., Kwak, H., & Hoteit, H. (2024). Numerical simulation of carbon dioxide flooding in fractured reservoirs using generic projection-based embedded discrete fracture model. Physics of Fluids, 36(10).

[21] Rao, X. (2023). A generic workflow of projection-based embedded discrete fracture model for flow simulation in porous media. Computational Geosciences, 27, 561–590.

[22] Rao, X., He, X., Kwak, H., & Hoteit, H. (2024). A hybrid method combining mimetic finite difference and discontinuous Galerkin for two-phase reservoir flow problems. International Journal for Numerical Methods in Fluids.

[23] Rao, X., Guo, S., He, X., Kwak, H., Yousef, A., & Hoteit, H. (2024). Hybrid mimetic finite difference and streamline methods for numerical simulation of two-phase flow in fractured reservoirs. Computers and Geotechnics, 166, 106048.

[24] Rao, X., Zhao, H., & Liu, Y. (2022). A meshless numerical modeling method for fractured reservoirs based on extended finite volume method. SPE Journal, 1–40.

[25] Rao, X., Xu, Y., & Liu, W. (n.d.). Meshless extended finite volume method for compositional reservoir models. Chinese Journal of Computational Mechanics, 1–9.

[26] Rao, X., Zhao, H., & Liu, Y. (2023). A novel meshless method based on the virtual construction of node control domains for porous flow problems. Engineering with Computers, 1–41.

[27] Rao, X. (2022). An upwind generalized finite difference method (GFDM) for meshless analysis of heat and mass transfer in porous media. Computational Particle Mechanics, 1–22.