Архітектурні рішення для масштабованих кластерних систем із підтримкою GPU

Громовський Олександр Володимирович [1]

Опубліковано	Секція	УДК
30.01.2024	Економіка	004.35:004.272

DOI: https://doi.org/10.5281/zenodo.17544024

Анотація. У статті розглянуто сучасні архітектурні рішення для побудови масштабованих кластерних систем із підтримкою графічних процесорів (GPU). Особливу увагу приділено динамічному розподілу ресурсів, гнучкому управлінню обчислювальними платформами, а також вдосконаленим механізмам планування з урахуванням локалізації даних. Наведено приклади реалізації високоефективних паралельних алгоритмів на основі CUDA, які демонструють значне зростання продуктивності при роботі з великими обсягами даних. Окреслено ключові виклики, пов'язані з гетерогенністю середовища, адаптацією програм до специфіки апаратної архітектури та реконфігурацією обладнання. Представлені підходи дають змогу підвищити ефективність обчислювальних кластерів у задачах наукових досліджень, моделювання та обробки даних.

Ключові слова: високопродуктивні обчислення; гетерогенне середовище; паралельна обробка даних; віртуалізація обчислювальних ресурсів; динамічне планування навантаження; обчислювальні прискорювачі; програмна адаптивність; локалізація даних; продуктивність CUDA-алгоритмів; оптимізація обчислювальних процесів.

Architectural solutions for scalable cluster systems with GPU support

Annotation. The paper examines modern approaches to the design and optimization of high-performance distributed infrastructures that utilize graphics accelerators. Particular attention is paid to strategies for dynamic resource management, hybrid execution environments, and data-aware scheduling techniques that aim to maximize computing efficiency. The study outlines the evolution of parallel processing models and highlights the role of virtualization, high-throughput memory, and energy-saving communication networks in building flexible infrastructures. The authors analyze implemented systems with thousands of GPUs and evaluate how the choice of interconnects and topologies impacts performance scaling. Software tools such as CUDA, OpenACC, NCCL, and orchestration platforms are discussed as core components enabling effective integration of heterogeneous nodes. The article demonstrates that a comprehensive architectural approach that combines high-speed networking, hardware specialization, and intelligent resource control leads to substantial improvements in performance and energy efficiency. The findings show that GPU-centric infrastructures can significantly reduce execution time and power consumption, especially in machine learning, simulation, and scientific research tasks. Challenges related to

[1] Senior Software Engineer, UAB IdeaMars, Lithuania

reconfiguration, hardware diversity, and software adaptation remain relevant and require further research.

Keywords: heterogeneous computing; parallel execution models; energy-aware systems; resource virtualization; dynamic load control; data locality optimization; container-based orchestration; performance profiling; high-throughput memory; scalable topologies.

Вступ

Постановка проблеми у загальному вигляді та її зв'язок із важливими науковими чи практичними завданнями. В умовах розвитку високопродуктивних обчислень (High Performance Computing, HPC) все більше наукових, інженерних та комерційних завдань потребують колосальної обчислювальної потужності. Із розвитком штучного інтелекту, глибинного навчання, обробки великих даних та моделювання складних фізичних систем, традиційні обчислювальні архітектури втрачають свою ефективність. У цьому контексті архітектура кластерних систем із підтримкою графічних процесорів (GPU) набула особливої актуальності. GPU, спочатку розроблені для комп'ютерної графіки, еволюціонували в потужні паралельні обчислювальні пристрої, що значно перевищують центральні процесори (CPU) в задачах паралельного типу [1].

Масштабовані кластерні системи з GPU підтримують гнучке нарощування обчислювальних ресурсів, що дозволяє адаптувати інфраструктуру до зростаючих потреб користувачів. Зважаючи на потребу в обчислювальній ефективності, мінімізації затримок, енергоефективності та забезпеченні масштабованості, архітектурні рішення в таких системах потребують спеціалізованого проєктування та досліджень.

Сучасні обчислювальні задачі в галузях біоінформатики, фізики високих енергій, кліматичного моделювання, фінансового прогнозування та машинного навчання вимагають мільйонів або навіть мільярдів операцій на секунду. Традиційні кластери, засновані на CPU, стикаються з обмеженнями масштабованості, енергоспоживання та обчислювальної щільності. Крім того, зі зростанням розміру задач збільшується й потреба в ефективному розподілі навантаження, міжвузловій комунікації та управлінні ресурсами [2, 3].

GPU-кластери дозволяють подолати частину цих обмежень, однак постають нові проблеми, зокрема архітектурного характеру: як ефективно інтегрувати GPU в обчислювальний кластер, як зменшити затримки при комунікації між CPU та GPU, як забезпечити балансування навантаження між вузлами з неоднорідною апаратною конфігурацією, а також як оптимізувати витрати енергії при збереженні продуктивності.

Ці завдання мають критичне значення не лише для академічних досліджень, а й для практичного застосування в промислових обчислювальних центрах, дата-центрах великих компаній та суперкомп'ютерах, що працюють на рівні національних інфраструктур.

Виділення невирішених раніше частин загальної проблеми. Попри значний прогрес у розвитку GPU-кластерів, залишаються відкритими питання ефективної інтеграції гібридних ресурсів, адаптивної маршрутизації даних, архітектурної уніфікації CPU-GPU-комунікацій, зменшення латентності при доступі до пам'яті та динамічного масштабування в умовах зміни навантаження.

Зокрема, невирішеною залишається проблема ефективного управління гетерогенною архітектурою, в якій GPU мають суттєво відмінні характеристики від CPU, що потребує спеціальних архітектурно-алгоритмічних підходів. Крім того, більшість кластерних ОС не забезпечують оптимальної підтримки GPU як повноцінного ресурсу, що обмежує їхню ефективність у багатокористувацькому середовищі [4].

Також недостатньо досліджені методи побудови кластерів з масштабованою мережею передачі даних, оптимізованою під великий обсяг GPU-GPU та GPU-CPU

взаємодій. Це викликає проблему вузьких місць у продуктивності навіть у кластерів із великою кількістю GPU, якщо не вирішено питання топології, розподілу задач та алгоритмічної взаємодії.

Формулювання цілей статті (постановка завдання). Метою даної статті є систематизація та узагальнення сучасних архітектурних рішень, які забезпечують масштабованість, ефективність та продуктивність кластерних обчислювальних систем з GPU-підтримкою.

Завдання статті. У межах поставленої мети в статті передбачено виконати наступні завдання:

1. проаналізувати сучасні архітектури кластерів з GPU;
2. охарактеризувати переваги та недоліки різних топологій об'єднання GPU у кластері;
3. дослідити особливості гібридного CPU-GPU керування задачами;
4. розглянути стратегії масштабування в залежності від характеру обчислювальних задач;
5. висвітлити приклади реалізованих систем та їх продуктивність.

Результати

Архітектурні рішення для масштабованих кластерних систем із підтримкою графічного процесора зосереджені на ефективному використанні декількох GPU у різних обчислювальних вузлах з метою підвищення загальної продуктивності системи та забезпечення обчислювальної ефективності на високому рівні. Такі рішення базуються на принципах динамічного розподілу ресурсів, удосконалених механізмах планування, а також на використанні гібридних моделей програмування, що дозволяють адаптуватися до вимог сучасних високопродуктивних додатків і масштабованих навантажень.

Одним із ключових підходів є реалізація динамічного розподілу ресурсів, що включає архітектури з приєднаними до мережі прискорювачами. У таких конфігураціях графічні процесори можуть динамічно призначатися до обчислювальних вузлів у режимі реального часу на основі поточних обчислювальних вимог, що значно покращує використання апаратних ресурсів. Ще одним прикладом є використання збірок GPGPU, де процесори та GPU об'єднуються у віртуальні платформи виконання. Це забезпечує гнучке управління ресурсами та безперервне відображення до апаратної частини системи, що у підсумку позитивно впливає на продуктивність і масштабованість [5].

Важливу роль у таких системах відіграють і розширені механізми планування. Одним із прикладів є впровадження алгоритмів планування, що враховують локалізацію даних. Таке «оповіщення про місцевість» дозволяє зменшити обсяги передаваних даних між вузлами та забезпечити ефективніше використання GPU, особливо у розподілених середовищах. У результаті цього суттєво скорочується загальний час виконання обчислювальних задач.

Крім цього, відзначається значне зростання продуктивності завдяки високій паралельній ефективності архітектур. Зокрема, впровадження алгоритмів кластеризації з використанням CUDA на GPU-кластерах дозволяє досягати прискорення виконання до 4794 разів порівняно з традиційними CPU-рішеннями. Такі результати підтверджують потенціал рішень, побудованих на графічних процесорах, для обробки великих обсягів даних та розв'язання складних задач машинного навчання й наукових розрахунків.

Утім, попри значні переваги, архітектурні рішення з GPU мають і певні обмеження. Однією з основних проблем залишається необхідність адаптації програмного забезпечення під специфічні конфігурації апаратного середовища. Крім того, складність у реконфігурації обладнання та гетерогенність сучасних обчислювальних

інфраструктур може впливати на ефективність масштабування, особливо в умовах змінних навантажень або змішаного використання різних типів процесорів [6].

Сучасна еволюція архітектур масштабованих кластерних систем із підтримкою GPU значно трансформувала уявлення про обчислювальні потужності та ефективність обробки великих даних. Існують десятки високопродуктивних систем, які демонструють різні підходи до реалізації архітектури GPU-кластерів. У таблиці 1 наведено відомі приклади таких систем, їх апаратну конфігурацію, типи міжвузлових з'єднань, рівень продуктивності та основні особливості реалізації. Це дозволяє простежити, як саме архітектурні рішення впливають на масштабованість та ефективність.

Таблиця 1

Приклади реальних систем із GPU

Назва системи	**Кількість GPU**	**Тип з'єднання**	**Продуктивність**	**Особливості**
NVIDIA Selene	2240	NVSwitch + HDR	>1 EFLOPS	DGX A100 + NCCL
ORNL Summit	27648	NVLink + IB	200 PFLOPS	IBM + NVIDIA

Наведені приклади демонструють різноманіття підходів до побудови масштабованих кластерів із GPU-підтримкою. Усі розглянуті системи використовують високошвидкісні мережі (наприклад, NVLink або InfiniBand) для забезпечення ефективної взаємодії між GPU та вузлами. Результати цих систем підтверджують, що оптимальна топологія, спеціалізоване обладнання та добре налаштоване ПЗ дозволяють досягти продуктивності світового рівня. При цьому використання GPU дозволяє не лише скоротити час виконання обчислень, але й зменшити енергоспоживання на одиницю продуктивності.

Розробка архітектурних моделей таких систем базується на інтеграції високопродуктивного апаратного забезпечення з інтелектуальним програмним керуванням, що дозволяє досягати високих показників продуктивності при оптимальному використанні ресурсів.

Архітектурна побудова кластерів з GPU передбачає використання багаторівневої організації апаратних компонентів. Функціонування кластерів з GPU неможливе без спеціалізованого програмного забезпечення, яке забезпечує розподіл ресурсів, взаємодію між компонентами та підтримку масштабованих обчислень. Таблиця 2 нижче узагальнює ключові програмні засоби, що використовуються в сучасних GPU-кластерах, із вказанням їхніх функцій та сфер застосування.

Таблиця 2

Програмне забезпечення для GPU-кластерів

Категорія	**Інструмент**	**Призначення**
Планування задач	SLURM, Torque	Керування чергами та GPU
Паралельне програмування	CUDA, MPI+Cuda, OpenACC	Написання високопродуктивного коду
Комунікаційні бібліотеки	NCCL, GDRCopy	Оптимізація міжGPU-взаємодії

Різноманітність інструментів у таблиці свідчить про необхідність комплексного підходу до організації обчислювального середовища. Кожен компонент — від планувальника задач до засобів міжпроцесорної комунікації — відіграє ключову роль у

досягненні високої продуктивності. Особливо важливою є сумісність програмного забезпечення з апаратною архітектурою, а також підтримка масштабування на тисячі GPU. Відсутність належного рівня автоматизації або оптимізації в будь-якому з цих інструментів може призвести до втрати обчислювальної ефективності.

На нижньому рівні відбувається глибока інтеграція GPU із центральним процесором (CPU) через високошвидкісні шини, такі як PCIe 4.0/5.0, а в новітніх системах — через NVLink та NVSwitch. Такий рівень взаємодії забезпечує мінімізацію затримок у передачі даних між компонентами і підвищує ефективність виконання паралельних обчислень. GPU сьогодні працюють не ізольовано, а формують обчислювальні блоки, здатні до самостійної обробки задач завдяки наявності великого обсягу пам'яті високої пропускної здатності (HBM2e, HBM3).

На середньому рівні архітектура кластерів організована як система обчислювальних вузлів, кожен з яких обладнаний кількома GPU. Ці вузли поєднуються у кластерну мережу з використанням інтерфейсів на зразок InfiniBand HDR/NDR або 100/200/400 Gigabit Ethernet. Саме мережа визначає ефективність міжвузлового обміну даними — одного з критичних факторів масштабованості. Надзвичайно важливим є питання топології: використання ієрархічних, тороподібних, деревовидних або повнозв'язних мереж дозволяє адаптувати архітектуру до характеру обчислювального навантаження.

На найвищому рівні архітектура GPU-кластерів передбачає реалізацію гнучкого програмного управління. Інфраструктурне програмне забезпечення включає такі компоненти, як системи оркестрації контейнерів (Docker, Kubernetes), планувальники задач (SLURM, PBS, Torque), засоби управління GPU-ресурсами (NVIDIA Docker, NVIDIA Data Center GPU Manager). Особливої уваги заслуговує підтримка масштабованих моделей програмування — CUDA, OpenCL, OpenACC, MPI+Cuda, що дозволяють реалізовувати як embarrassingly parallel, так і складні обчислювальні графи зі зваженим розподілом навантаження.

У роботі [7] представлено архітектурні рішення для реалізації інтелектуального чисельного програмного забезпечення на кластерних MIMD-системах. Зокрема, розглянуто досвід створення й оптимізації програмних модулів для комп'ютера Інпарком з урахуванням ефективної взаємодії процесорів і балансування навантаження, що має значення для сучасних кластерів із підтримкою GPU.

Дослідження реальних сценаріїв використання масштабованих кластерів засвідчує, що архітектурні рішення безпосередньо впливають на продуктивність. Наприклад, при використанні NVSwitch всередині обчислювального вузла досягається надзвичайно низька латентність комунікацій між GPU, що критично для задач навчання великих мовних моделей (LLM) або тривимірного моделювання. У кластері NVIDIA Selene, побудованому з DGX A100 систем, вдалося досягти понад 1 ЕФОПС (exaFLOP) при виконанні змішаних обчислень завдяки архітектурі, орієнтованій на глибоке з'єднання GPU. У той час як у системах на зразок Summit (ORNL), реалізовано ієрархічний підхід, де вузли з GPU з'єднані через NVLink, а міжвузлове з'єднання здійснюється через InfiniBand, що дозволяє забезпечити масштабування до сотень тисяч ядер.

Окремим напрямом результатів є аналіз ефективності програмного забезпечення у гібридних архітектурах. Наприклад, використання NVIDIA NCCL дозволяє автоматично оптимізувати колективні операції комунікації (broadcast, reduce, all-to-all), зменшуючи вплив мережевих затримок на загальний час виконання. В умовах зростання числа GPU в кластері надзвичайно важливою є здатність масштабувати ці операції без лінійного збільшення часу.

Ще однією важливою складовою архітектурних рішень є енергоспоживання. У багатьох суперкомп'ютерах GPU використовуються не лише завдяки їхній продуктивності, але і через вищу енергоефективність на одиницю FLOPS. Дослідження

енергетичного профілю кластерів показує, що використання динамічного керування частотою GPU, активація енергозберігаючих режимів у моменти простою, оптимізація мережевого стека значно впливають на загальне енергоспоживання. В умовах обмеженого енергобюджету такі архітектурні рішення є критичними для побудови стійких екосистем HPC.

Важливим етапом аналізу є порівняння ефективності різних архітектур за кількісними показниками. Таблиця 3 демонструє ключові метрики, зокрема продуктивність на одиницю енергії, пропускну здатність і масштабованість систем. Ці параметри мають вирішальне значення при виборі архітектури для високопродуктивних задач і є основою для прийняття інженерних рішень.

Таблиця 3

Метрики ефективності архітектур

Показник	CPU-кластер	GPU-кластер	Коментар
FLOPS/Вт	~0.1	>0.6	GPU в 5–6 разів ефективніші
Пропускна здатність	Середня	Висока	NVSwitch та InfiniBand лідирують
Масштабованість	Обмежена	Висока	Особливо при колективних задачах

Метрики, наведені в таблиці, наочно демонструють перевагу GPU-орієнтованих архітектур у ключових параметрах, що визначають ефективність обчислень. Значно вища енергоефективність та пропускна здатність роблять GPU-кластери незамінними для задач високої складності. Масштабованість також залишається сильною стороною таких систем, особливо при використанні розподілених паралельних обчислень. Проте досягнення максимальних показників можливе лише за умови комплексної оптимізації всієї архітектури — від апаратного рівня до програмного стека.

Узагальнюючи результати, можна зазначити, що оптимальна архітектура масштабованого кластера з GPU базується на глибокій інтеграції апаратної платформи, програмного середовища і топології передачі даних. Саме їхнє узгодження дозволяє досягти як масштабованості, так і стабільної продуктивності. Практичні реалізації доводять, що правильно сконструйована архітектура зменшує витрати часу на обробку задач у 10–50 разів порівняно з CPU-кластерами та забезпечує економію енергії до 40%.

Для кращого розуміння переваг кластерів із GPU-підтримкою порівняємо базові характеристики різних типів обчислювальних архітектур. У таблиці 4 нижче наведено порівняння традиційних CPU-кластерів, спеціалізованих GPU-кластерів та гібридних систем, що поєднують обидва типи процесорів. Це дозволяє оцінити доцільність впровадження тієї чи іншої архітектури залежно від задач.

Таблиця 4

Порівняльна таблиця архітектур вузлів

Характеристика	CPU-кластер	GPU-кластер (NVLink)	Гібрид (CPU+GPU)
Продуктивність	Низька-середня	Висока	Дуже висока
Енергоефективність	Низька	Висока	Висока
Складність реалізації	Низька	Висока	Висока

Як видно з таблиці, GPU-кластери забезпечують суттєво вищу продуктивність, хоча потребують складнішого проектування та налаштування. Гібридні системи демонструють найкращий баланс між гнучкістю обробки різнорідних задач та

енергоефективністю, проте вимагають складних програмних рішень для ефективного розподілу навантаження між CPU і GPU.

При цьому варто відмітити, що незважаючи на стрімкий розвиток високопродуктивних архітектур із підтримкою GPU, традиційні кластери на основі центральних процесорів залишаються важливою складовою сучасної обчислювальної інфраструктури. Їх релевантність зберігається передусім у контексті задач із послідовним характером обчислень, де паралелізм має обмежене застосування або навіть не є бажаним з точки зору ефективності та коректності обробки даних.

Центральні процесори спроектовані як універсальні обчислювальні пристрої, здатні виконувати широкий спектр інструкцій і ефективно працювати зі складними послідовними логічними структурами, гілкуванням, залежностями між даними та операціями вводу-виводу. У таких умовах GPU, які орієнтовані на масивно паралельні обчислення з мінімальними залежностями, втрачають свою перевагу. Наприклад, багато типів алгоритмів — зокрема, обробка текстів, де серійність визначає результат, криптографічні розрахунки з багатьма умовними переходами або симуляції з часовою залежністю — значно краще реалізуються саме на CPU.

Ще однією перевагою CPU-кластерів є гнучка підтримка складних програмних стеків і операційних систем. На відміну від GPU, які часто потребують спеціальних драйверів, фреймворків та обмеженого середовища, CPU-системи дозволяють запускати широкий спектр застосунків із мінімальними адаптаціями. Це особливо важливо в наукових проєктах із довгостроковим життєвим циклом програмного забезпечення, яке розроблено десятки років тому і потребує стабільного послідовного виконання.

Крім того, CPU демонструють кращу продуктивність у завданнях, де критичною є взаємодія з оперативною пам’яттю, зокрема при роботі з великими базами даних, коли доступ до дисків або кешів відбувається ітеративно або з непередбачуваними патернами. Завдяки складній системі кешування, префетчингу (prefetching) та оптимізованому доступу до пам’яті, CPU-кластери дозволяють забезпечити стабільну пропускну здатність навіть у непаралелізованих фрагментах коду.

Слід зазначити й нижчий поріг входу при розробці та налагодженні програм під CPU. Традиційні мови програмування, як-от C/C++, Fortran, Python, мають повну підтримку на CPU-платформах, що дає змогу швидко створювати та масштабувати нові проєкти без необхідності вивчення специфікацій GPU, таких як CUDA або OpenCL.

З технічної точки зору, CPU-кластери простіші в адмініструванні та масштабуванні для базових потреб. Вони не потребують спеціальних систем охолодження, енергозабезпечення та високошвидкісних з’єднань між GPU, що часто ускладнює побудову великих кластерів із графічними прискорювачами. Такі системи є більш енергоекономними для нескладних задач і мають широку підтримку з боку популярних систем розподілених обчислень, як-от OpenMPI, Hadoop, Spark тощо.

Варто наголосити, що більшість реальних додатків містять як паралельні, так і послідовні компоненти. Саме CPU-кластери є ідеальним середовищем для обробки контрольних структур, послідовного вводу-виводу, логіки управління, роботи з файловою системою та іншими процесами, які не масштабуються ефективно на GPU. Тому навіть у гібридних архітектурах роль CPU залишається центральною — вони керують загальним виконанням, координують ресурси та виконують ті частини алгоритмів, де GPU неефективні.

Таким чином, традиційні CPU-кластери залишаються релевантними для задач із послідовними обчисленнями, але значно поступаються у масштабованості.

Висновки

Враховуючи тенденції розвитку високопродуктивних обчислювальних систем, стає очевидним, що майбутнє належить масштабованим кластерним архітектурам з

підтримкою GPU. Їхнє впровадження дозволяє суттєво підвищити ефективність обробки даних у різноманітних наукових та прикладних сферах — від штучного інтелекту до молекулярного моделювання та кліматичних симуляцій.

У даній статті було представлено комплексний огляд архітектурних рішень, які забезпечують функціонування та масштабованість кластерів із GPU. Детально розглянуто принципи апаратної організації, мережевої взаємодії, програмного управління та забезпечення енергоефективності. Показано, що тільки узгоджене поєднання цих компонентів дозволяє досягти оптимальних результатів у системах, що масштабуються до тисяч і десятків тисяч GPU.

Проведений аналіз також виявив низку напрямів для подальших досліджень: це і розробка нових алгоритмів балансування навантаження з урахуванням гетерогенності апаратури, і створення інтелектуальних систем моніторингу енергоспоживання, і побудова динамічних архітектур із можливістю адаптації до змін задач у реальному часі. Особливої уваги заслуговують питання автоматизації конфігурації кластерів із GPU через застосування методів машинного навчання, що дозволяє зменшити участь людини у керуванні складними обчислювальними системами.

Таким чином, архітектурні рішення для масштабованих кластерних систем із підтримкою GPU є не лише технічною платформою для високопродуктивних обчислень, але й рушієм еволюції цифрової науки, промислових інновацій та технологій майбутнього. Їхнє подальше вдосконалення має вирішальне значення для розвитку обчислювальної інфраструктури в глобальному масштабі.

Список використаних джерел

1. Hazra A., Rana P., Adhikari M., Amgoth T. Fog computing for next generation internet of things: fundamental, state-of-the-art and research challenges. Computer Science Review. 2023. Vol. 48. P. 100549.
2. Klenk B. *Communication Architectures for Scalable GPU-centric Computing Systems*. 2018. https://hgpu.org/?p=17946
3. Малярчук Р. В. Архітектура багатокластерної системи на базі мікросервісу синхронізації. Київ, 2021. - 107 с.
4. Samizadeh I. A brief introduction to two data processing architectures–Lambda and Kappa for Big Data / Iman Samizadeh// Medium. Towards Data Science. [Електронний ресурс] Режим доступу: URL: https://towardsdatascience.com/a-brief-introduction-to-two-data-processing architectures-lambda-and-kappa-for-big-data-4f35c28005bb
5. Srirama S. N. A decade of research in fog computing: relevance, challenges, and future directions. Software: Practice and Experience. 2024. Vol. 54(1). P. 3–23.
6. Oden L. Direct communication between distributed GPUs. Doctoral Dissertation, Ruprecht-Karls University Heidelberg, Mannheim, Germany, 2014.
7. Химич А.Н., Молчанов И.Н, Попов А.В., Чистякова Т.В., Яковлев М.Ф., Громовский А.В. Интеллектуальное численное программное обеспечение MIMD-компьютера Инпарком. *Искусственный интеллект,* №3, 2007 г. С. 436-446