УДК 004.032.26

К. С. КУРОЧКА, Ю. С. БАШАРИМОВ, Ю. Д. ЁВЖЕНКО

СТРАТЕГИИ ПАРАЛЛЕЛИЗМА КАК КЛЮЧЕВОЙ ФАКТОР РАЗВЕРТЫВАНИЯ LARGE LANGUAGE MODELS НА БАЗЕ ПОТРЕБИТЕЛЬСКИХ GPU

Гомельский государственный технический университет им. П. О. Сухого
г. Гомель, Республика Беларусь

Аннотация. *Экспоненциальный рост размеров больших языковых моделей (LLM) создает существенные барьеры для их локального развертывания, обусловленные нехваткой видеопамяти (VRAM) на одиночных устройствах. Целью работы является выявление и обоснование наиболее эффективной стратегии параллелизма для инференса LLM на кластерах из потребительских графических процессоров (GPU), объединенных медленной шиной PCIe. Методы исследования включали проведение серии вычислительных экспериментов для сравнения монолитной архитектуры (NVIDIA RTX A6000) и распределенной системы (2x NVIDIA RTX 3090) с использованием фреймворка vLLM. Анализировалось влияние тензорного (Tensor Parallelism) и конвейерного (Pipeline Parallelism) параллелизма на ключевые метрики: пропускную способность, задержку (TTFT, TPOT) и стабильность энергопотребления при запуске модели DeepSeek-R1-Distill-Llama-14B. Результаты однозначно указывают на непригодность тензорного параллелизма для систем без NVLink из-за критических задержек синхронизации. Доказано, что конвейерный параллелизм является единственной жизнеспособной стратегией для PCIe-кластеров, обеспечивая высокую пропускную способность, несмотря на наличие периодов простоя («пузырей») и менее стабильный профиль энергопотребления по сравнению с монолитным решением. В заключении сформулированы рекомендации по использованию мульти-GPU конфигураций: они являются оптимальным экономическим выбором для задач, критичных к объему памяти, таких как Retrieval-Augmented Generation (RAG), позволяя масштабировать VRAM значительно дешевле профессиональных аналогов.*

Ключевые слова: *большие языковые модели, LLM, инференс, тензорный параллелизм, конвейерный параллелизм, vLLM, GPU, CUDA, NVLink, PCIe, RAG*

Введение

В последние годы Large Language Models (LLM) стали движущей силой в области искусственного интеллекта, открывая новые горизонты для обработки естественного языка [1] и автоматизации сложных предметных областей, таких как генерация формализованных структур из естественного языка [2]. Однако экспоненциальный рост их размеров – с миллиардов до сотен миллиардов параметров [3] – создал серьезные проблемы для рядовых пользователей, исследователей и небольших компаний [4]. Главным «узким местом» в этом процессе стала видеопамять (VRAM), поскольку для эффективной работы с моделью она должна быть загружена целиком в память графического процессора (GPU).

Проблема нехватки вычислительных ресурсов (VRAM) особенно остро стоит для образовательных учреждений с ограниченным бюджетом на инфраструктуру [5].

В условиях ограниченного бюджета или при наличии уже имеющегося оборудования, возникает закономерный вопрос: что лучше – приобрести одну дорогостоящую, высокопроизводительную видеокарту с большим объемом VRAM, или использовать две менее мощные, суммарная стоимость которых может быть ниже, а затраты на электричество будут нивелированы. Если в мире компьютерных игр технология SLI/NVLink для объединения двух GPU оказалась нишевым решением, то для задач, связанных с LLM, ситуация иная.

Существуют три основные стратегии параллелизма: по данным, тензорный и конвейерный, однако в условиях ограниченной VRAM на потребительских GPU наиболее актуальны последние две.

Тензорный параллелизм (TP) разделяет веса модели внутри слоев между несколькими GPU, базируясь на принципах архитектуры Megatron-LM [6], позволяя преодолеть ограничения VRAM [7]. Однако его эффективность критически зависит от интерконнекта. Разрыв в пропускной способности между NVLink (600–900 ГБ/с) и PCIe 4.0 (~64 ГБ/с) является решающим фактором [8]: при использовании PCIe время передачи данных начинает доминировать над вычислениями, что особенно заметно на этапе prefill.

Конвейерный параллелизм (PP) снижает нагрузку, передавая данные только на границах блоков слоев, аналогично подходу GPipe [9]. Его главный недостаток – «пузыри» простоя (bubbles) на начальном и конечном этапах конвейера. В иерархии масштабирования TP используется для локальных сверхбыстрых соединений (NVLink), тогда как PP и параллелизм по данным (DP) применяются для межблочных и межкластерных связей с меньшей пропускной способностью.

Тестирование

Для тестирования использован фреймворк vLLM [10]. Реализованная в нем технология PagedAttention управляет KV-кэшем по принципу виртуальной памяти, устраняя фрагментацию и обеспечивая утилизацию VRAM до 96 %, что критично для потребительских видеокарт.

Стоит подробнее остановиться на механизме управления памятью, так как он играет ключевую роль в эффективности Pipeline Parallelism. Традиционные системы инференса резервируют непрерывные блоки памяти под максимальную длину контекста (например, 4096 или 8192 токена), что неизбежно приводит к внутренней фрагментации: если запрос пользователя короче зарезервированного буфера, остальная память простаивает («memory waste»). На потребительских GPU с ограниченным объемом VRAM (24 ГБ) это критично.

Используемая технология PagedAttention решает эту проблему, разбивая KV-кэш на блоки фиксированного размера, которые могут располагаться в физической памяти несмежно, аналогично страницам в операционных системах. Для конфигурации конвейерного параллелизма это означает более плотную упаковку микро-батчей. В нашей экспериментальной установке модель была разделена на стадии (stages) с передачей тензоров активаций (activations) размером Batch×Sequence×Hidden_Size только на границах групп слоев. В отличие от TP, где передача данных происходит тысячи раз за один проход, PP требует всего одной передачи данных на микро-батч между видеокартами, что драматически снижает требования к пропускной способности шины PCIe и позволяет нивелировать ее низкую скорость.

Сравнение проводилось между монолитной системой (1x NVIDIA RTX A6000, 48 ГБ) и распределенной конфигурацией (2x NVIDIA RTX 3090, суммарно 48 ГБ). Связка RTX 3090 обладает вдвое большей вычислительной мощностью (20992 против 10752 CUDA-ядер), однако обмен данными между ними ограничен шиной PCIe (без NVLink). Тестовая модель – DeepSeek-R1-Distill-Llama-14B [11].

Конфигурация 2x RTX 3090 тестировалась в двух режимах: тензорный (tensor_parallel_size=2) и конвейерный (pipeline_parallel_size=2) параллелизм. Это позволяет оценить влияние медленного интерконнекта (PCIe) на распределенные вычисления. Измеряемые метрики: пропускная способность (Throughput), время до первого токена (TTFT), скорость генерации (TPOT) и энергопотребление.

Анализ пропускной способности (рисунок 1) подтверждает неэффективность тензорного параллелизма на шине PCIe: частые синхронизации приводят к критическому падению скорости. Напротив, режим конвейерного параллелизма (PP) успешно нивелирует задержки интерконнекта. В сравнении с монолитной RTX A6000, связка 2x RTX 3090 в режиме PP показывает более высокую производительность на всех размерах контекста, эффективно используя удвоенную вычислительную мощность (20992 ядра) несмотря на отсутствие NVLink.

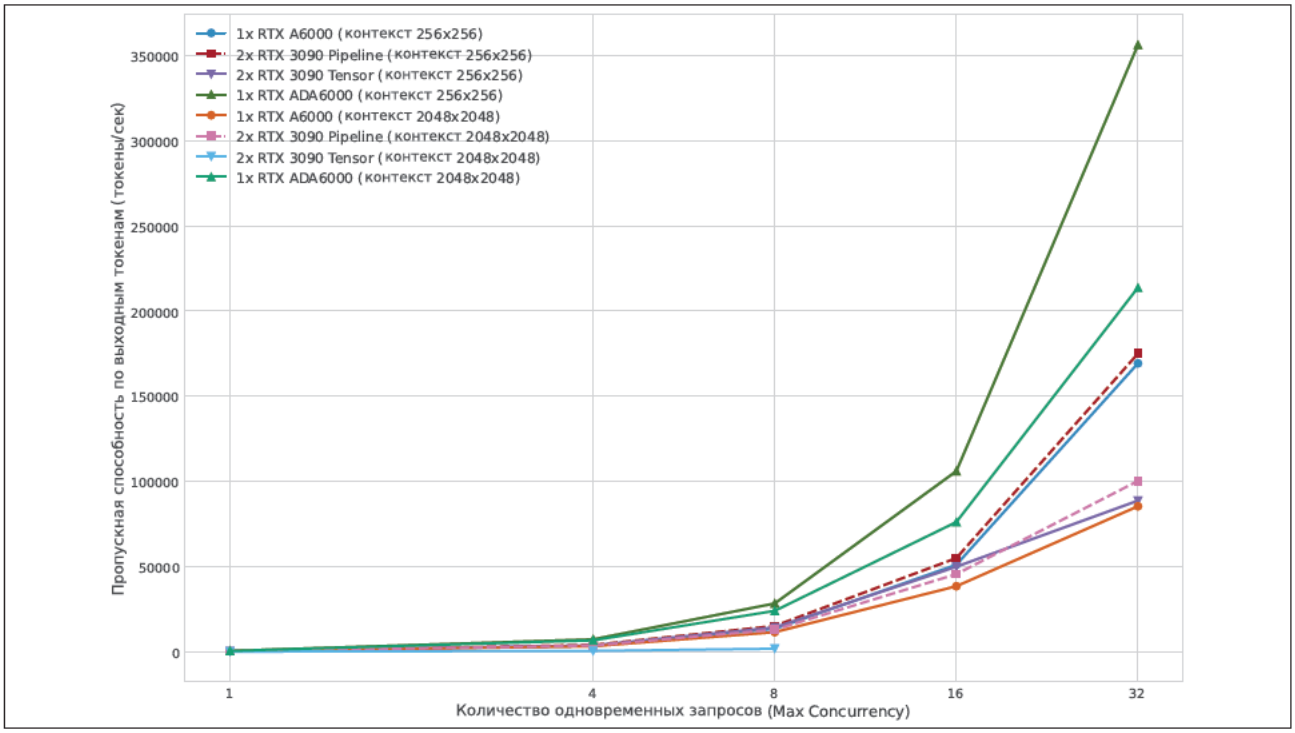


Рисунок 1. Сравнение пропускной способности (токены/сек) в зависимости от количества одновременных запросов

Анализ TTFT (рисунок 2) подтверждает неэффективность тензорного параллелизма на PCIe из-за высоких задержек этапа prefill. В отличие от него, конвейерный режим (PP) демонстрирует отзывчивость, сопоставимую с монолитной RTX A6000 при нагрузке до 16 запросов, так как передача данных происходит лишь на границах стадий.

Однако в точке насыщения (32 запроса) накладные расходы на межпроцессорную коммуникацию в PP приводят к более резкому росту задержки по сравнению с архитектурой A6000, свободной от сетевых издержек.

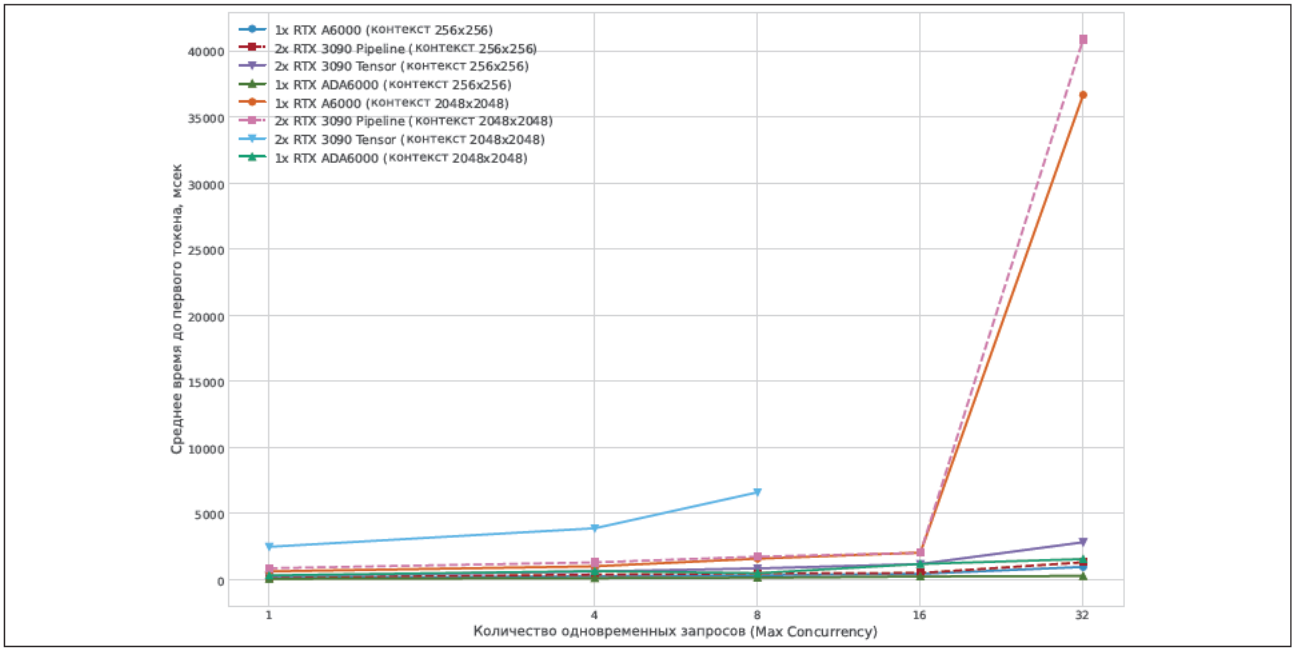


Рисунок 2. Сравнение времени до первого токена (TTFT, мс) в зависимости от нагрузки

Анализ TPOT (рисунок 3) фиксирует задержку генерации свыше 350 мс для тензорного параллелизма, вызванную синхронизацией каждого токена через PCIe. Конвейерный режим устраняет это ограничение: благодаря параллельной обработке разных запросов в батче, межпроцессорная коммуникация перестает быть блокирующим фактором, обеспечивая высокую производительность на этапе decode.

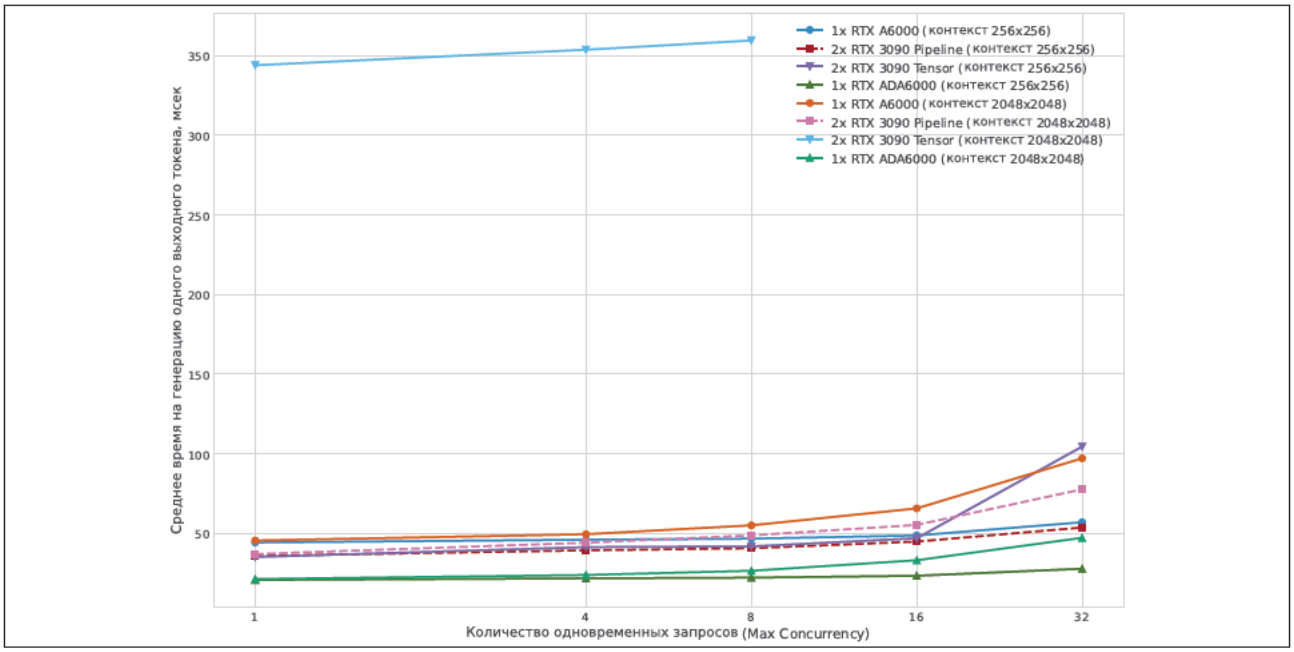


Рисунок 3. Сравнение скорости генерации токенов (TPOT, мс) в зависимости от нагрузки

Анализ энергопотребления (рисунок 4) демонстрирует преимущество монолитной архитектуры. График A6000 показывает стабильную утилизацию на уровне лимита 300 Вт. Напротив, система 2x RTX 3090, достигая пиков 370–380 Вт, имеет выраженный нестабильный профиль. Характерные «провалы» мощности визуализируют проблему «пузырей» (bubbles) в конвейере – периоды простоя GPU в ожидании данных. Эти накладные расходы на синхронизацию снижают итоговую энергоэффективность распределенной системы по сравнению с непрерывной работой одиночной карты.

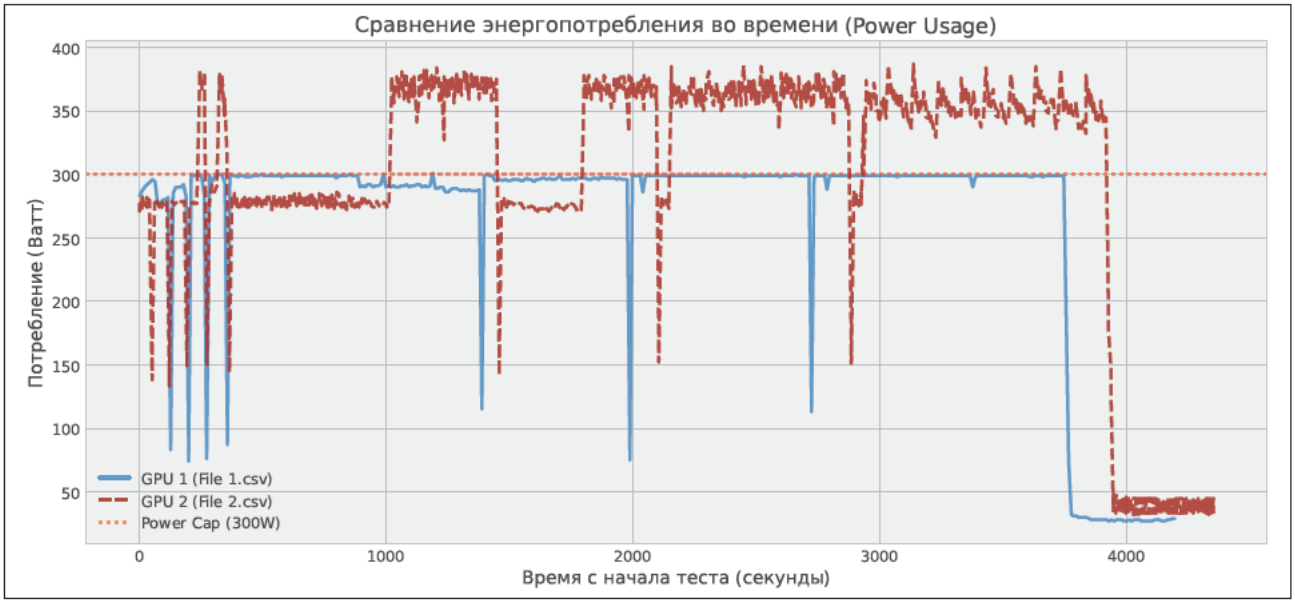


Рисунок 4. Сравнение энергопотребления во времени между одиночной видеокартой A6000 и системой из двух RTX 3090

Заключение

Результаты исследования показывают, что при отсутствии высокоскоростного соединения NVLink выбор стратегии распараллеливания становится определяющим фактором производительности. Тензорный параллелизм в условиях ограниченной пропускной способности шины PCIe демонстрирует критическое падение эффективности из-за высоких задержек синхронизации, тогда как конвейерный подход (Pipeline Parallelism) подтвердил свою жизнеспособность за счет минимизации частоты межпроцессорных обменов. Сравнительный анализ архитектур выявил, что монолитная система (RTX A6000) обеспечивает лучшую энергоэффективность и непрерывную утилизацию ресурсов. Распределенная конфигурация из двух потребительских видеокарт (RTX 3090), несмотря на двукратное превосходство в теоретической вычислительной мощности, демонстрирует снижение итоговой производительности из-за периодов простоя («пузырей») в конвейере и накладных расходов на передачу данных. Тем не менее данная связка является экономически оптимальным решением для задач, критичных к объему видеопамяти, таких как Retrieval-Augmented Generation (RAG) [12] и пакетная обработка длинного контекста, предлагая существенно лучшее соотношение цены и доступного объема VRAM по сравнению с профессиональными решениями.

ЛИТЕРАТУРА

1. Attention is all you need / A. Vaswani, N. Shazeer, N. Parmar [et al.] // Advances in Neural Information Processing Systems. 2017. Vol. 30. DOI:10.48550/arXiv.1706.03762.

2. Курочка, К. С. Технология трансляции естественно-языковых правил мерчандайзинга в цифровые планограммы / К. С. Курочка, Ю. Д. Ёвженко // Информатика. 2025. Т. 22, № 4. С. 55–64. DOI: 10.37661/1816-0301-2025-22-4-55-64.

3. Scaling laws for neural language models / J. Kaplan, S. McCandlish, T. Henighan [et al.] // arXiv preprint arXiv:2001.08361. 2020. DOI: 10.48550/arXiv.2001.08361.

4. Large Language Models: A survey / S. Minaee, T. Mikolov, N. Nikzad [et al.] // arXiv preprint arXiv:2402.06196. 2024. DOI: 10.48550/arXiv.2402.06196.

5. Курочка, К. С. Нейросетевая модель автоматической генерации тестовых заданий для студентов в системе Moodle на основе анализа методических материалов / К. С. Курочка, Ю. С. Башаримов // Цифровая трансфор-

мация. 2025. Т. 31, № 3. С. 66–75. DOI: 10.35596/1729-7648-2025-31-3-66-75.

6. Megatron-LM: Training multi-billion parameter language models using model parallelism / M. Shoeybi, M. Patwary, R. Puri [et al.] //arXiv preprint arXiv:1909.08053. 2019. DOI: 10.48550/arXiv.1909.08053.

7. Efficiently Scaling Transformer Inference / R. Pope, S. Douglas, A. Chowdhery [et al.] // Proceedings of Machine Learning and Systems (MLSys 2023). 2023. Vol. 5. P. 606–624. URL: https://proceedings mlsys.org/paper_files/paper/2023/hash/c4be71ab8d24cdfb45e3d06dbfca2780-Abstract-mlsys2023.html (date of access: 10.02.2026).

8. PowerInfer: Fast Large Language Model serving with a consumer-grade GPU / Y. Song, Z. Mi, H. Xie, H. Chen // Proceedings of the ACM SIGOPS 30th Symposium on Operating Systems Principles (SOSP'24). 2024. P. 590-606. DOI: 10.1145/3694715.3695964.

9. GPipe: Efficient training of giant neural networks using pipeline parallelism / Y. Huang, Y. Cheng, A. Bapna [et al.] // Advances in Neural Information Processing Systems (NeurIPS 2019). 2019. Vol. 32. URL: https://proceedings.neurips.cc/papers/search?q=GPipe%3A+Efficient+training+of+giant+neural+networks+using+pipeline+parallelism+ (date of access: 10.02.2026).

10. Efficient memory management for Large Language Model serving with PagedAttention / W. Kwon, Z. Li, S. Zhuang [et al.] // Proceedings of the 29th Symposium on Operating Systems Principles (SOSP'23). 2023. P. 611–626. DOI: 10.1145/3600006.3613165.

11. DeepSeek-V2: A strong, economical, and efficient Mixture-of-Experts language model / A. Liu, B. Feng, B. Wang [et al.] // arXiv preprint arXiv:2405.04434. 2024. DOI: 10.48550/arXiv.2405.04434.

12. Retrieval-augmented generation for knowledge-intensive NLP tasks / P. Lewis, E. Perez, A. Piktus [et al.] // Advances in Neural Information Processing Systems. (NeurIPS 2020). 2020. Vol. 33. URL: https://proceedings.neurips.cc/papers/search?q=Retrieval-Augmented+Generation+for+Knowledge-Intensive+NLP+Tasks+ (date of access: 10.02.2026).

REFERENCES

1. Vaswani A., Shazeer N., Parmar N., Uszkoreit J., Jones L., Gomez A.N., et al. Attention is all you need. Advances in Neural Information Processing Systems. 2017;30. https://doi.org/10.48550/arXiv.1706.03762.

2. Kurochka K.S., Youzhanka Y.D. The technology of translating natural language merchandising rules into digital planograms. Informatics. 2025;22(4):55−64 (in Russian). http://dx.doi.org/10.37661/1816-0301-2025-22-4-55-64.

3. Kaplan J., McCandlish S., Henighan T., Brown T.B., Chess B., Child R., et al. Scaling laws for neural language models. arXiv preprint arXiv:2001.08361. 2020. https://doi.org/10.48550/arXiv.2001.08361.

4. Minaee S., Mikolov T., Nikzad N., Chenaghlu M., Socher R., Amatriain X., et al. Large Language Models: A survey. arXiv preprint arXiv:2402.06196. 2024. https://doi.org/10.48550/arXiv.2402.06196.

5. Kurochka K.S., Basharymau Y.S. Neural network model for automated test generation for students in the Moodle system based on the analysis of methodological materials. Digital Transformation. 2025;31(3):66–75 (in Russian). http://dx.doi.org/10.35596/1729-7648-2025-31-3-66-75.

6. Shoeybi M., Patwary M., Puri R., Le Gresley P., Casper J., Catanzaro B. Megatron-LM: Training multi-billion parameter language models using model parallelism. arXiv preprint arXiv:1909.08053. 2019. https://doi.org/10.48550/arXiv.1909.08053.

7. Pope R., Douglas S., Chowdhery A., Devlin J., Bradbury J., Heek J., et al. Efficiently scaling transformer inference. Proceedings of Machine Learning and Systems (MLSys 2023). 2023;(5):606–624. Available at: https://proceedings mlsys.org/paper_files/paper/2023/hash/c4be71ab8d24cdfb45e3d06dbfca2780-Abstract-mlsys2023.html (accessed 10 February 2026).

8. Song Y., Mi Z., Xie H., Chen H. Powerinfer: Fast Large Language Model serving with a consumer-grade GPU. Proceedings of the ACM SIGOPS 30th Symposium on Operating Systems Principles (SOSP'24). 2024:590–606. https://doi.org/10.1145/3694715.3695964.

9. Huang Y., Cheng Y., Bapna A., Firat O., Chen D., Chen M., et al. GPipe: Efficient training of giant neural networks using pipeline parallelism. Advances in Neural Information Processing Systems (NeurIPS 2019). 2019;32. Available at: https://proceedings neurips.cc/papers/search?q=GPipe%3A+Efficient+training+of+giant+neural+networks+using+pipeline+parallelism+ (accessed 10 February 2026).

10. Kwon W., Li Z., Zhuang S., Sheng Y., Zheng L., Yu C.H., et al. Efficient memory management for Large Language Model serving with PagedAttention. Proceedings of the 29th Symposium on Operating Systems Principles (SOSP'23). 2023:611–626. https://doi.org/10.1145/3600006.3613165.

11. Liu A., Feng B., Wang B., Wang B., Liu B., Zhao C., et al. DeepSeek-V2: A strong, economical, and efficient Mixture-of-Experts language model. arXiv preprint arXiv:2405.04434. 2024. https://doi.org/10.48550/arXiv.2405.04434.

12. Lewis P., Perez E., Piktus A., Petroni F., Karpukhin V., Goyal N., et al. Retrieval-augmented generation for knowledge-intensive NLP tasks. Advances in Neural Information Processing Systems (NeurIPS 2020). 2020;33. Available at: https://proceedings neurips.cc/papers/search?q=Retrieval-Augmented+Generation+for+Knowledge-Intensive+NLP+Tasks+ (accessed 10 February 2026).

K. S. KUROCHKA, Y. S. BASHARYMAU, Y. D. YOUZHANKA

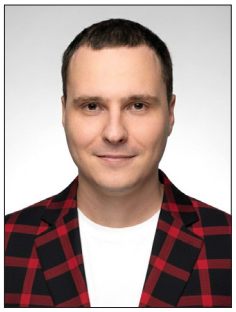
PARALLELISM STRATEGIES AS A KEY FACTOR FOR DEPLOYING LARGE LANGUAGE MODELS ON CONSUMER GPUS

Sukhoi State Technical University of Gomel
Gomel, Republic of Belarus

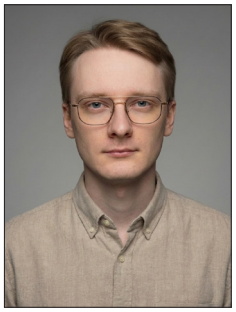
Abstract. *The exponential growth in the size of Large Language Models (LLMs) creates significant barriers to their local deployment, primarily due to Video RAM (VRAM) shortages on single devices. The aim of this work is to identify and substantiate the most effective parallelism strategy for LLM inference on consumer Graphics Processing Unit (GPU) clusters connected via a slow PCIe bus. Research methods included a series of experiments comparing a monolithic architecture (NVIDIA RTX A6000) and a distributed system (2x NVIDIA RTX 3090) using the vLLM framework. The impact of Tensor Parallelism (TP) and Pipeline Parallelism (PP) on key metrics – throughput, latency (TTFT, TPOT), and power consumption stability – was analyzed while running the DeepSeek-R1-Distill-Llama-14B model. The results unequivocally indicate the unsuitability of Tensor Parallelism for systems without NVLink due to critical synchronization delays. It is proven that Pipeline Parallelism is the only viable strategy for PCIe clusters, ensuring high throughput despite the presence of idle periods («bubbles») and a less stable power consumption profile compared to the monolithic solution. In conclusion, recommendations for using multi-GPU configurations are formulated: they represent the optimal economic choice for memory-critical tasks, such as Retrieval-Augmented Generation (RAG), allowing VRAM scaling at a significantly lower cost than professional analogs.*

Keywords: *Large Language Models, LLM, inference, Tensor Parallelism, Pipeline Parallelism, vLLM, GPU, CUDA, NVLink, PCIe, RAG*

Курочка Константин Сергеевич
Гомельский государственный технический университет им. П. О. Сухого (г. Гомель, Республика Беларусь). Кандидат технических наук, доцент, заведующий кафедрой «Информационные технологии».
Konstantin S. Kurochka
Sukhoi State Technical University of Gomel (Gomel, Republic of Belarus). PhD in Engineering, Associate Professor, Head of the Department of Information Technologies.
E-mail: kurochka@gstu.by



Башаримов Юрий Сергеевич
Гомельский государственный технический университет им. П. О. Сухого (г. Гомель, Республика Беларусь). Ассистент кафедры «Информационные технологии».
Yury S. Basharymau
Sukhoi State Technical University of Gomel (Gomel, Republic of Belarus). Assistant Lecturer at the Department of Information Technologies.
E-mail: basharymauyury@gmail.com



Ёвженко Юрий Дмитриевич
Гомельский государственный технический университет им. П. О. Сухого (г. Гомель, Республика Беларусь). Магистрант кафедры «Информационные технологии».
Yury D. Youzhanka
Sukhoi State Technical University of Gomel (Gomel, Republic of Belarus). Master's Student at the Department of Information Technologies.
E-mail: yuevzhenko@gmail.com