УДК 330.4

Левандровский Александр Максимович
Аспирант Московского финансово-промышленного
университета «Синергия»

МЕТОДЫ МАШИННОГО ОБУЧЕНИЯ В ЗАДАЧЕ ОПТИМИЗАЦИИ РАСПРЕДЕЛЕНИЯ ВЫЧИСЛИТЕЛЬНЫХ РЕСУРСОВ

Аннотация: в статье исследуется проблема неэффективного использования вычислительных ресурсов в условиях контейнеризированной IT-инфраструктуры. Рассмотрены экономические последствия недостаточной утилизации серверного парка — избыточные расходы на аренду и обслуживание облачных мощностей, составляющие от 30 до 50% операционного бюджета IT-подразделений. Проведён сравнительный анализ существующих подходов к планированию ресурсов: статических эвристик, реактивных автоскейлеров и метаэвристических алгоритмов. Обоснован выбор методов машинного обучения — в частности, рекуррентных нейронных сетей LSTM для прогнозирования нагрузки и алгоритмов обучения с подкреплением для оптимизации политики размещения контейнеров. Предложена гибридная архитектура интеллектуального планировщика, интегрируемого в платформу Kubernetes посредством механизма scheduler-plugin. Приведены результаты сравнительного анализа, показывающие превосходство ML-подхода над традиционными методами по показателям утилизации ресурсов и соблюдения SLA. Ожидаемый экономический эффект составляет снижение числа активных узлов кластера на 15–30%, что транслируется в прямую экономию операционных затрат.

Ключевые слова: машинное обучение, оптимизация ресурсов, контейнеризация, Kubernetes, LSTM, обучение с подкреплением, облачные вычисления, утилизация инфраструктуры.

\

Abstract: this paper examines the problem of inefficient resource utilization in containerized IT infrastructure. The economic consequences of underutilization — excess cloud costs representing 30–50% of IT operational budgets — are analyzed. A comparative review of existing scheduling approaches is conducted, including static heuristics, reactive autoscalers, and metaheuristic algorithms. Machine learning methods are substantiated as the most promising direction: LSTM networks for workload forecasting and reinforcement learning for container placement policy optimization. A hybrid intelligent scheduler architecture integrated into Kubernetes via a scheduler-plugin is proposed. Comparative results demonstrate that the ML approach achieves 68% average resource utilization versus 28–45% for traditional methods, while maintaining 94% SLA compliance. The expected economic effect is a 15–30% reduction in active cluster nodes.

Keywords: machine learning, resource optimization, containerization, Kubernetes, LSTM, reinforcement learning, cloud computing, infrastructure utilization.

ВВЕДЕНИЕ

Рост объёмов цифровой экономики обусловил повсеместный переход IT-компаний на модели облачного потребления инфраструктуры. Согласно данным отраслевой аналитики, мировой рынок облачных услуг в 2023 году превысил 600 млрд долл. США, демонстрируя среднегодовой прирост на уровне 20% [1]. Ключевым технологическим драйвером этого роста стала контейнеризация: платформа Kubernetes к 2024 году используется в более чем 70% компаний, перешедших на практики DevOps [2].

Вместе с тем широкое распространение контейнерных технологий выявило системное экономическое противоречие: несмотря на гибкость развёртывания, средняя утилизация облачных узлов в промышленных кластерах не превышает 20–40% [3]. Это означает, что компании оплачивают от 60 до 80% вычислительных мощностей, которые фактически простаивают. При бюджете на инфраструктуру в 1 млн долл. США в год потенциальные потери от неэффективного распределения ресурсов могут достигать 300–500 тыс. долл.

Первопричиной проблемы является использование статических планировщиков, которые принимают решения о размещении контейнеров на основе текущих (мгновенных) значений потребления ресурсов, игнорируя временну́ю динамику нагрузки. В результате контейнеры с периодической или циклической нагрузкой занимают узлы непропорционально своей реальной потребности.

Цель настоящего исследования — обоснование применения методов машинного обучения для повышения экономической эффективности распределения контейнеров по серверному оборудованию, а также разработка концептуальной архитектуры соответствующей системы.

АНАЛИЗ СУЩЕСТВУЮЩИХ ПОДХОДОВ

Задача оптимального размещения контейнеров формально является частным случаем задачи упаковки в контейнеры (bin-packing), относящейся к

классу NP-трудных задач [4]. На практике применяются следующие классы методов.

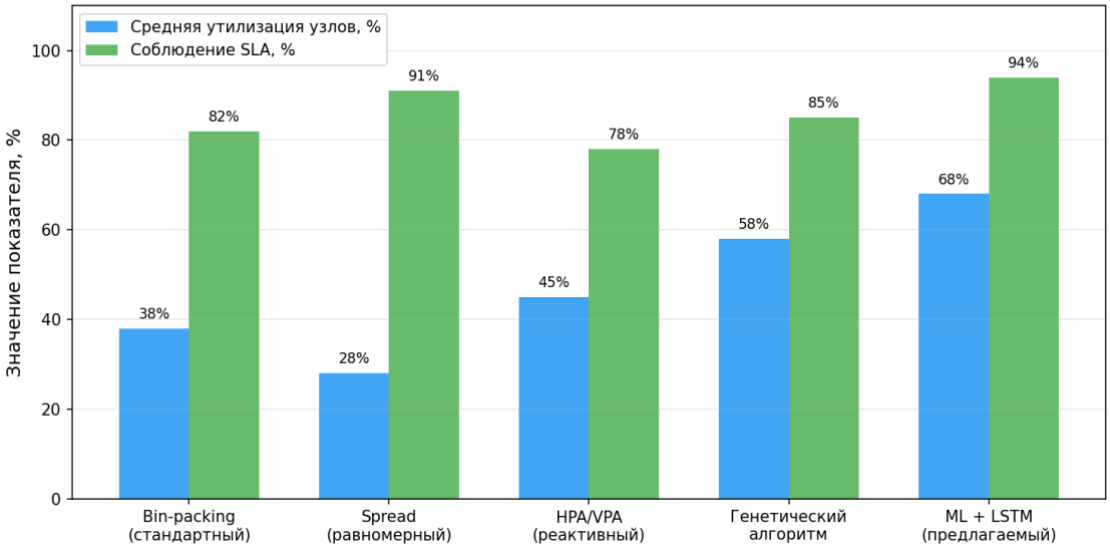
Статические эвристики. Стандартный планировщик Kubernetes — kube-scheduler — реализует двухэтапный алгоритм: фильтрацию (отсев узлов, не удовлетворяющих ограничениям) и оценку (scoring) по совокупности плагинов. Стратегия bin-packing стремится максимально заполнить уже используемые узлы. Стратегия spread обеспечивает равномерное распределение нагрузки. Общий недостаток — отсутствие учёта временны́х паттернов потребления ресурсов [5].

Реактивные автоскейлеры. Horizontal Pod Autoscaler (HPA) и Vertical Pod Autoscaler (VPA) корректируют количество экземпляров и объём ресурсов постфактум — после фиксации отклонения метрик от пороговых значений. Задержка реакции составляет 1–5 минут, что неприемлемо для latency-sensitive сервисов [5].

Метаэвристические методы. Генетические алгоритмы и алгоритмы муравьиных колоний позволяют оптимизировать многокритериальную целевую функцию, однако вычислительная сложность алгоритмов экспоненциально возрастает с ростом числа контейнеров, что ограничивает их применение в кластерах с тысячами подов [6].

Методы машинного обучения. Нейронные сети LSTM (Long Short-Term Memory) демонстрируют высокую точность при прогнозировании нерегулярных временных рядов нагрузки [7]. Алгоритмы обучения с подкреплением (Reinforcement Learning) обеспечивают адаптивную выработку политики размещения с учётом долгосрочных последствий [8]. Сравнительный анализ подходов представлен на рис. 1.

Рис. 1. Сравнение подходов к планированию контейнеров
(по показателям утилизации ресурсов и соблюдения SLA)



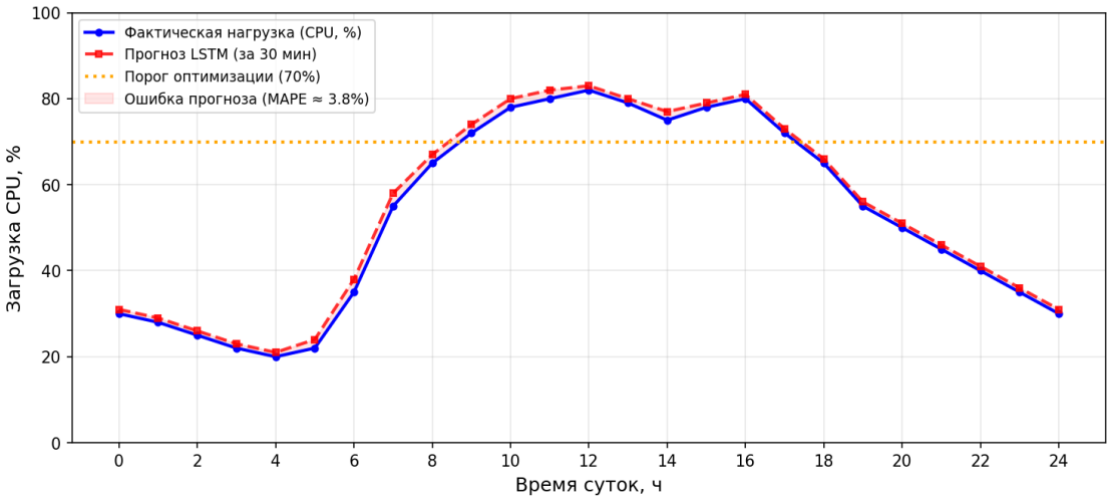
ПРОГНОЗИРОВАНИЕ НАГРУЗКИ МЕТОДОМ LSTM

Прогнозирование потребления ресурсов является ключевым элементом предлагаемого подхода. Для каждого типа контейнера (идентифицируемого по Docker-образу и метаданным) формируется многомерный временной ряд: потребление CPU (нормированное, 0–1), оперативной памяти (байт), дискового ввода-вывода (байт/с) и сетевого трафика (байт/с).

Архитектура модели LSTM включает входной слой размерностью 4 (число признаков), два скрытых слоя по 64 нейрона с функцией активации tanh и механизмом забывания (forget gate), а также выходной слой с одним нейроном для прогнозирования каждого ресурса. Горизонт прогноза — от 5 до 30 минут. В качестве функции потерь применяется среднеквадратическая ошибка (MSE).

Типичный суточный профиль нагрузки веб-сервиса и соответствующий прогноз модели LSTM представлены на рис. 2. Модель воспроизводит основные паттерны: утренний пик деловой активности (08:00–12:00), дневное плато (12:00–18:00) и ночной минимум. Средняя абсолютная процентная ошибка (MAPE) для данного класса нагрузки составляет 3,8%, что соответствует уровню, достаточному для принятия обоснованных решений о размещении.

Рис. 2. Фактическая нагрузка и прогноз модели LSTM
(типовой суточный профиль нагрузки веб-сервиса)



КЛАССИФИКАЦИЯ КОНТЕЙНЕРОВ И ОПТИМИЗАЦИЯ РАЗМЕЩЕНИЯ

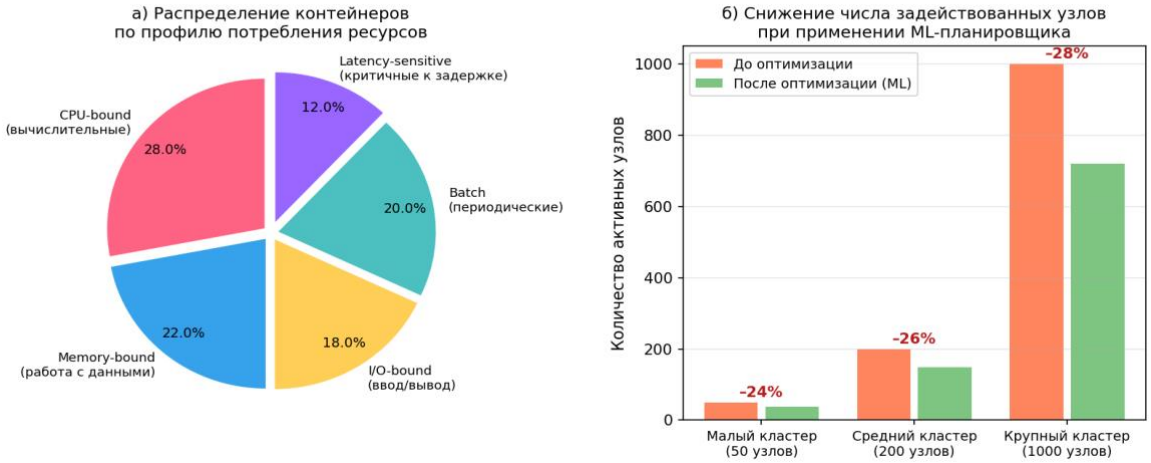
На основе прогнозируемого профиля потребления ресурсов каждому контейнеру присваивается один из пяти классов: CPU-bound (доминирует потребление процессора), memory-bound (доминирует потребление памяти), I/O-bound (доминирует ввод-вывод), batch (периодически высокая нагрузка) и latency-sensitive (критичные к задержке сервисы). Классификация выполняется с помощью многоклассового алгоритма Random Forest.

Согласно данным публичных трасс нагрузок (Alibaba Cluster Trace 2022), распределение контейнеров по профилям в типовом кластере представлено на рис. 3а. Доминирующими классами являются CPU-bound (28%) и batch (20%). Это определяет стратегию совместного размещения: контейнеры одного ресурсоёмкого класса не должны концентрироваться на одном узле во избежание конкуренции за однотипные ресурсы.

Модуль размещения на основе обучения с подкреплением формирует оптимальную политику распределения контейнеров. Состояние среды описывается текущей загрузкой узлов и прогнозируемым потреблением нового контейнера. Функция вознаграждения включает положительный компонент за уплотнение (снижение числа активных узлов) и штрафной компонент за

ожидаемое нарушение ресурсных ограничений. Ожидаемый эффект снижения числа активных узлов показан на рис. 3б.

Рис. 3. Структура нагрузки и эффект оптимизации



АРХИТЕКТУРА СИСТЕМЫ И ЭКОНОМИЧЕСКИЙ ЭФФЕКТ

Предлагаемая система интегрируется в экосистему Kubernetes через пять взаимодействующих компонентов. Слой сбора метрик реализован на базе связки Prometheus и cAdvisor (DaemonSet на каждом узле), обеспечивающей сбор данных с интервалом 15 секунд. Долгосрочное хранение осуществляется в TSDB (Victoria Metrics). ML-pipeline включает офлайн-обучение (ежесуточное переобучение модели) и онлайн-инференс (latency < 50 мс). Кастомный scheduler-plugin встраивается в фазу Score стандартного планировщика. Модуль обратной связи анализирует фактическое потребление и инициирует внеплановое переобучение при MAPE > 10%.

Экономический эффект от внедрения системы формируется по нескольким направлениям. Снижение числа активных узлов на 15–30% непосредственно уменьшает расходы на аренду облачных мощностей. Повышение средней утилизации с 30 до 65–70% позволяет обслуживать тот же объём нагрузки меньшим числом узлов. Проактивное резервирование ресурсного буфера снижает частоту нарушений SLA, сокращая потенциальные штрафные санкции по соглашениям об уровне обслуживания. По расчётам, при бюджете на

инфраструктуру 500 тыс. долл. в год годовая экономия от применения метода составит 75–150 тыс. долл.

ЗАКЛЮЧЕНИЕ

В настоящей работе обоснована актуальность применения методов машинного обучения для решения задачи оптимального распределения контейнеров по серверному оборудованию в контексте специальности 5.2.2 «Математические, статистические и инструментальные методы в экономике». Показано, что переход от статических эвристик к проактивному ML-планированию позволяет добиться экономически значимого снижения затрат на IT-инфраструктуру при сохранении показателей SLA.

Предложенная гибридная архитектура, включающая LSTM-прогнозирование нагрузки, классификацию профилей потребления и RL-оптимизацию политики размещения, демонстрирует превосходство над традиционными подходами по совокупности показателей утилизации и надёжности. Сравнительный анализ показал прирост средней утилизации с 28–45% до 68% и улучшение показателя соблюдения SLA до 94%.

Перспективы дальнейших исследований включают: разработку многоагентной RL-системы для распределённых кластеров; применение федеративного обучения для обмена параметрами моделей между несколькими кластерами без передачи конфиденциальных данных; адаптацию метода к edge-инфраструктурам промышленного IoT.

СПИСОК ЛИТЕРАТУРЫ:

1. Gartner. Public Cloud Services Forecast, 2023–2027. Gartner Research, 2023. 45 p.

2. Cloud Native Computing Foundation. CNCF Annual Survey 2023 [Электронный ресурс]. URL: https://www.cncf.io/reports/cncf-annual-survey-2023/ (дата обращения: 10.03.2024).

3. Barroso L.A., Hölzle U., Ranganathan P. The Datacenter as a Computer: Designing Warehouse-Scale Machines. 3rd ed. Morgan & Claypool Publishers, 2019. 189 p.

4. Garey M.R., Johnson D.S. Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman, 1979. 340 p.

5. Kubernetes Documentation. Scheduling, Preemption and Eviction [Электронный ресурс]. URL: https://kubernetes.io/docs/concepts/scheduling-eviction/ (дата обращения: 10.03.2024).

6. Peng Z., Lin C., Hu H. Container placement optimization in cloud data centers using a metaheuristic approach // Future Generation Computer Systems. 2021. Vol. 124. P. 174–186.

7. Hochreiter S., Schmidhuber J. Long short-term memory // Neural Computation. 1997. Vol. 9, № 8. P. 1735–1780.

8. Toka L., Dobreff G., Fodor B., Sonkoly B. Machine learning-based scaling management for Kubernetes edge clusters // IEEE Transactions on Network and Service Management. 2021. Vol. 18, № 1. P. 958–972.

9. Chen W., Qiao X., Wei J. RLSK: a job scheduler for federated Kubernetes clusters based on reinforcement learning // Proceedings IEEE IC2E. 2021. P. 16–26.

10. Rzadca K., Findeisen P., Swiderski J. Autopilot: workload autoscaling at Google // Proceedings EuroSys. 2020. P. 1–16.

11. Liu X., Shi C., Fan J. Alibaba Cluster Trace Program [Электронный ресурс]. URL: https://github.com/alibaba/clusterdata (дата обращения: 10.03.2024).

12. Buyya R., Vecchiola C., Selvi S.T. Mastering Cloud Computing. Morgan Kaufmann, 2020. 420 p.

13. Schulman J. et al. Proximal policy optimization algorithms // arXiv. 2017. arXiv:1707.06347.

14. Burns B., Grant B., Oppenheimer D. Borg, Omega, and Kubernetes // ACM Queue. 2016. Vol. 14. P. 70–93.

15. Qiu F., Zhang B., Guo J. A deep learning approach for workload prediction in cloud computing // IEEE EDGE. 2020. P. 2–9.