УДК 004.052.32

СКООРДИНИРОВАННЫЙ СБРОС НАГРУЗКИ КАК МЕХАНИЗМ ОБЕСПЕЧЕНИЯ УСТОЙЧИВОСТИ МИКРОСЕРВИСНЫХ СИСТЕМ ПРИ ПЕРЕГРУЗКАХ

Астахова К.А. (ИТМО)
Научный руководитель – инженер Зубаков А.Н. (ИТМО)

Введение. Микросервисная архитектура стала стандартом при разработке крупномасштабных облачных приложений, обеспечивая гибкость, масштабируемость и ускорение разработки. Однако переход от монолитных систем к распределённым графам сервисов радикально усложнил обеспечение надёжности и производительности [1]. Один пользовательский запрос теперь проходит через десятки сервисов, формируя глубокие и разветвлённые графы вызовов с множеством зависимостей. В таких условиях перегрузка становится основной причиной сбоев [1]. Практика эксплуатации крупных систем (Steam, Spotify, Coinbase и др.) показывает, что традиционные механизмы защиты — таймауты, ограничители скорости, предохранители и автоматическое масштабирование — не предотвращают каскадные сбои при резких всплесках нагрузки [2]. Ключевая проблема заключается не в том, когда отклонять запросы, а в том, какие запросы следует обслуживать в условиях дефицита ресурсов. В этой связи особый интерес представляет механизм сброса нагрузки (load shedding), предполагающий проактивное отклонение части запросов для сохранения общей устойчивости и выполнения целевых показателей уровня сервиса (SLO) [2].

Основная часть. В работе проведён аналитический обзор современных академических и промышленных подходов к сбросу нагрузки в микросервисных системах: DAGOR, Breakwater, TopFull, Rajomon, а также решений Netflix, AWS и Uber Cinnamon [1-6]. Показано, что существующие подходы можно классифицировать по четырём ключевым критериям: охват координации по графу вызовов, поддержка бизнес-приоритетов, гранулярность управления и время реакции на перегрузку. Академические фреймворки (DAGOR, Rajomon, TopFull) демонстрируют высокую эффективность за счёт координации между сервисами и раннего отклонения запросов, что позволяет существенно увеличить полезную пропускную способность и снизить хвостовые задержки [1, 4, 5]. Однако они в недостаточной степени учитывают бизнес-критичность различных типов запросов и сложны во внедрении в реальных системах. Промышленные решения (Netflix, AWS, Uber) напротив, строятся вокруг бизнес-приоритетов и простоты эксплуатации, обеспечивая быструю реакцию на перегрузки [2, 6], но действуют преимущественно локально, без учёта полной структуры графа вызовов, что приводит к частичной обработке запросов и неэффективному расходованию ресурсов. На основе сравнительного анализа выявлен ключевой разрыв между исследовательскими и промышленными подходами: отсутствие децентрализованного, скоординированного механизма сброса нагрузки, учитывающего бизнес-приоритеты на уровне интерфейсов и способного работать в реальном времени.

Выводы. Проведённый анализ показал, что повышение устойчивости микросервисных систем при перегрузках требует объединения преимуществ академических и промышленных подходов. Перспективным направлением является разработка децентрализованного механизма сброса нагрузки, который:

– координирует решения по всему графу вызовов;

– учитывает бизнес-приоритеты запросов;

– обеспечивает раннее отклонение избыточной нагрузки;

– согласован с существующими механизмами устойчивости (таймауты, retry, circuit breaker, autoscaling).

Результаты работы могут быть использованы при проектировании систем управления нагрузкой в распределённых сервисных архитектурах, а также служат основой для разработки прототипа адаптивного механизма сброса нагрузки и его экспериментальной оценки.

Список использованных источников

1. Zhou H. et al. Overload control for scaling wechat microservices //Proceedings of the ACM Symposium on Cloud Computing. – 2018. – С. 149-161.
2. Keeping Netflix reliable using prioritized load shedding [Электронный ресурс] // Netflix TechBlog. – 2020. – URL: https://netflixtechblog.com/keeping-netflix-reliable-using-prioritizedload-shedding-6cc827b02f94 (дата обращения: 14.12.2025).
3. Cho I. et al. Overload control for {µs-scale}{RPCs} with breakwater //14th USENIX Symposium on Operating Systems Design and Implementation (OSDI 20). – 2020. – С. 299-314.
4. Park J. et al. TopFull: An Adaptive Top-Down Overload Control for SLOOriented Microservices //Proceedings of the ACM SIGCOMM 2024 Conference. – 2024. – С. 876-890.
5. Xing J. et al. Rajomon: Decentralized and Coordinated Overload Control for {Latency-Sensitive} Microservices //22nd USENIX Symposium on 25 Networked Systems Design and Implementation (NSDI 25). – 2025. – С. 21-36.
6. Cinnamon: using century-old tech to build a mean load shedder [Электронный ресурс] // Uber Blog. – 2025. – URL: https://www.uber.com/en-LT/blog/cinnamon-using-century-old-tech-tobuild-a-mean-load-shedder/ (дата обращения: 14.12.2025).