ISSN: 3030-3346

# МОДЕЛИРОВАНИЕ МНОГОУРОВНЕВОЙ РАСПРЕДЕЛЕННОЙ ПРОГРАММНОЙ АРХИТЕКТУРЫ IOT НА ОСНОВЕ ЦЕПОЧКИ ОПЕРАТОРОВ

*Жолдасбаев Даўронбек Рустем ули*

*Ташкентский университет информационных технологий имени Мухаммада ал-Хоразмий*
*E-mail: d.joldasbaev@tuit.uz*

*K E Y W O R D S*

IoT; цепочка операторов; многоуровневая архитектура; граничные вычисления; туманные вычисления; адаптивная передача данных; распределение вычислительных задач; балансировка нагрузки

*A B S T R A C T*

Предложена формальная модель цепочки операторов для управления потоками данных в распределённых IoT-системах. На её основе синтезирована многоуровневая архитектура Sensor–Edge–Fog–Cloud, объединяющая сбор, фильтрацию, адаптивную передачу, анализ и распределение задач. Экспериментальная оценка показала снижение сетевого трафика на 30–35 %. Коэффициенты использования вычислительных ресурсов и сбалансированности нагрузки составили 0,78 и 0,94 соответственно.

## Введение

Технологии Интернета вещей (IoT) широко используются в агропромышленном секторе для управления микроклиматом теплиц, мониторинга почвенных и водных ресурсов, а также мониторинга состояния животных в животноводческих комплексах [1,2,3]. В таких системах десятки и сотни датчиков непрерывно генерируют данные. Это приводит к росту сетевого трафика и нагрузки на центральные серверы [4]. В результате возникают существенные ограничения по задержке, энергопотреблению и надежности системы [5].

Одним из эффективных методов уменьшения избыточной передачи данных в агропромышленных системах IoT является адаптивное управление интервалом передачи в соответствии с изменчивостью параметров датчиков. В предыдущем исследовании был разработан алгоритм адаптивного управления интервалом передачи для мониторинга теплицы. Экспериментальная оценка показала, что увеличение интервала в стабильных условиях и его сокращение при изменении параметров позволяют снизить энергопотребление сенсорных узлов [6]. Однако данный алгоритм решает только задачу управления передачей данных. Его интеграция с другими этапами обработки потока данных и многоуровневой вычислительной инфраструктурой требует отдельной формальной модели.

Для снижения задержки и сетевой нагрузки обработку данных распределяют между уровнями Edge, Fog и Cloud [7–10]. Современные исследования охватывают предварительную обработку и фильтрацию на Edge [11,12], адаптивное размещение вычислительных задач и балансировку нагрузки [13–15], а также децентрализованное управление [16,17]. Однако полный цикл обработки данных от сбора до визуализации и согласованное размещение программных компонентов по уровням Sensor–Edge–Fog–Cloud недостаточно формализованы, что требует единой модели межуровневой интеграции.

Рассмотренные работы посвящены отдельным аспектам распределенной инфраструктуры IoT: обработке и фильтрации данных, распределению задач и балансировке нагрузки [7–17]. Вместе с тем полный цикл обработки данных – от сбора до визуализации – обычно не рассматривается в рамках единой формальной модели. Недостаточно систематизирован и вопрос согласованного размещения программных компонентов по уровням Sensor–Edge–Fog–Cloud. Это обусловливает необходимость разработки

(cc) BY 

целостной модели, формализующей межуровневую интеграцию и позволяющей адаптировать архитектуру к различным объектам IoT.

Цель исследования состоит в разработке модели цепочки операторов, объединяющей процессы управления информационными потоками агропромышленных IoT-систем. На основе этой модели выполняется синтез многоуровневой архитектуры программного комплекса.

**Модель цепочки операторов**

В агропромышленной системе IoT используется модель цепочки операторов для представления процессов от формирования данных с датчиков до представления конечных результатов пользователю в виде единой формальной структуры. Модель определяется как упорядоченный набор из пяти операторов, соответствующих основным функциональным этапам потока данных:

$$(M = \langle S, F, T, G, V \rangle) \quad (1)$$

где $S$ – сбор данных, $F$ – фильтрация, $T$ – управление передачей, $G$– анализ и распределение задач, а $V$ – оператор визуализации результатов. Каждый оператор определяется как функция, которая сопоставляет соответствующее входное множество с выходным множеством.

**Оператор сбора данных *(S)*** – принимает исходные данные датчиков и формирует упорядоченный по временным меткам поток данных. Если $D_{raw}$ – набор необработанных данных датчика, а $X$ – набор временных рядов, то оператор сбора данных определяется следующим отображением:

$$S: D_{raw} \rightarrow X \quad (2)$$

Каждое измерение датчика снабжается временной меткой и идентификатором источника, после чего передается на следующий этап обработки.

**Оператор фильтрации *(F)*** – удаляет повторяющиеся и малоинформативные значения на основе оценки статистической изменчивости данных на уровне Edge. Адаптивный порог фильтрации для каждого $i$-го датчика определяется следующим образом:

$$\theta_i = \lambda \cdot \sigma_i \quad (3)$$

где $\sigma_i$ – среднеквадратичное отклонение значений $i$-го датчика в выбранном временном окне, $\lambda>0$ – коэффициент корректировки, определяющий чувствительность фильтрации.

Оператор $F$ отображает набор исходных временных рядов $X$ на отфильтрованный набор данных $F:X\rightarrow Y$. Адаптивный порог позволяет сократить объем избыточных данных, передаваемых на следующий этап обработки, с учетом их текущей изменчивости.

**Оператор передачи *(T)*** – управляет интервалом передачи отфильтрованных данных на вышележащие уровни архитектуры.

Адаптивный механизм работы оператора был разработан в предыдущем исследовании авторов и экспериментально оценен на основе данных мониторинга теплицы [6]. В этом механизме интервал передачи динамически изменяется в диапазоне $\Delta t_{\min}$ и $\Delta t_{\max}$ в зависимости от текущего уровня изменения наблюдаемых параметров.

$$\Delta x_j(t) = \left| x_j(t) - x_j(t_{last}) \right|,$$

$$\Delta t_{k+1} = \begin{cases} min\left(\Delta, t_k + \delta, \Delta t_{max}()_j(t)_j\right) \\ max\left(\Delta, t_k - \delta, \Delta t_{min}()_j(t)_j\right) \end{cases} \quad (4)$$

Здесь $x_j(t)$ – текущее значение $j$-го мониторингового параметра; $x_j(t_{last})$ – значение на момент последней передачи; $\theta_j$ – пороговое значение изменения, установленное для этого параметра; и $\delta$ – шаг изменения интервала передачи. В стабильных условиях интервал передачи постепенно увеличивается, а при обнаружении значительного изменения параметров интервал уменьшается. В результате количество избыточных передач ограничивается при сохранении информативности данных. Оператор $T$ сопоставляет отфильтрованный набор данных с переданным набором данных: $T:Y\rightarrow Z$.

(cc) BY 

Для частного случая применения оператора $T$ при мониторинге теплицы использовались следующие параметры: $x_j=\{T,H\}$, $\theta_T=1^oC$, $\theta_H=5\%$, $\Delta t_{min}=10$ *мин*., $\Delta t_{max}=60$ *мин*. и $\delta=5$ *мин.* [6]

**Оператор анализа и распределения задач** *(G)* – координирует использование вычислительных ресурсов уровней Fog и Cloud. Он анализирует поступающие данные и распределяет вычислительные задачи между доступными узлами. Пусть $Q$ – множество задач, которые необходимо выполнить, а $R$ – текущее состояние ресурсов узлов Edge, Fog и Cloud. Тогда оператор определяется отображением $G:Z\times Q\times R\rightarrow W$. Где $W$ – множество результатов распределения вычислительных задач и анализа данных. Для распределения задач используется многокритериальная целевая функция, которая одновременно учитывает задержку, энергопотребление, сетевую нагрузку и дисбаланс нагрузки вычислительных узлов:

$$J = \alpha_1\tilde{L} + \alpha_2\tilde{E} + \alpha_3\tilde{N} + \alpha_4\tilde{I} \rightarrow min \qquad (5)$$
$$\sum_{k=1}^{4}\alpha_k = 1,\ \alpha_k \geq 0.$$

Здесь $\tilde{L}$, $\tilde{E}$, $\tilde{N}$ и $\tilde{I}$ – нормированные показатели задержки, энергопотребления, сетевой нагрузки и дисбаланса нагрузки соответственно; $\alpha_k$ – весовые коэффициенты, определяющие относительную важность критериев. Нормализация позволяет совместно учитывать в единой целевой функции критерии, имеющие различные единицы измерения.

**Оператор визуализации** *(V)* – представляет результаты анализа и распределения задач, сформированные оператором *G*, через пользовательский интерфейс. Оператор определяется отображением $V:W\rightarrow U$. Где $U$ – множество данных, предоставляемых пользователю, включая результаты мониторинга и анализа, а также сведения о состоянии системы.

Целостное функционирование цепочки операторов выражается через их функциональную композицию. Предполагая, что значения $Q$ и $R$ заданы для текущей вычислительной среды, вводится обозначение $G_{Q,R}(Z)=G(Z,Q,R)$. В таком случае общее отображение от сырых сенсорных данных до результата, предоставляемого пользователю, выражается следующим образом:

$$U = \left(V \circ G_{Q,R} \circ T \circ F \circ S\right)(D_{raw}) \qquad (6)$$

Для корректного определения композиции образ каждого оператора должен содержаться в области определения следующего оператора:

$$\begin{aligned} &Im(S) \subseteq Dom(F),\\ &Im(F) \subseteq Dom(T),\\ &Im(T) \subseteq Dom\left(G_{Q,R}\right),\\ &Im\left(G_{Q,R}\right) \subseteq Dom(V). \end{aligned} \qquad (7)$$

Условие (7) формально задает согласованность последовательных этапов обработки данных в цепочке операторов. Логическая модель цепочки операторов связывается с физической инфраструктурой IoT посредством проекционного отображения. Пусть множество сетевых уровней определено как

$$\ell = \{Sensor, Edge, Fog, Cloud\}.$$

Проекционное отображение, определяющее множество уровней выполнения каждого оператора, задается как:

$$\pi: \{S, F, T, G, V\} \rightarrow 2^{\ell} \qquad (8)$$

$$\begin{aligned} &\pi(S) = \{Sensor\},\\ &\pi(F) = \{Edge\},\\ &\pi(T) = \{Edge\},\\ &\pi(G) = \{Fog, Cloud\},\\ &\pi(V) = \{Cloud\}. \end{aligned} \qquad (9)$$

Эта проекция связывает логические функции цепочки операторов с физическими вычислительными уровнями, которые их реализуют, и обеспечивает формальную основу для синтеза многоуровневой программной архитектуры.

Проекция цепочки операторов на сетевые уровни представлена в таблице 1. При этом

(cc) BY 

каждый оператор привязан к основному уровню, на котором он работает.

**Таблица 1**

**Проекция цепочки операторов на уровни IoT-архитектуры**

| Оператор | Основной уровень | Выполняемая функция | Выход |
|---|---|---|---|
| *S* – сбор данных | Sensor | Сбор исходных данных датчиков и формирование временных рядов с добавлением временных меток | *X* – временные ряды |
| *F* – фильтрация | Edge | Адаптивная фильтрация повторяющихся и малоинформативных значений $(\theta_i=\lambda\cdot\sigma_i)$ | *Y* – отфильтрованный поток |
| *T* – передача | Edge | Адаптивное управление интервалом передачи на основе изменчивости параметров датчиков и передача потока *Z* на уровень Fog [6] | *Z* – переданная информация |
| *G* – анализ и распределение задач | Fog / Cloud | Распределение и анализ задач на основе многокритериальной целевой функции *J* | *W* – результаты распределения задач и анализа |
| *V* – визуализация | Cloud | Представление результатов в пользовательском интерфейсе | *U* – информация для пользователя |

**Синтез многоуровневой программной архитектуры**

На основе модели (1)–(7) и проекции (8)–(9) синтезирована многоуровневая архитектура Sensor–Edge–Fog–Cloud (табл. 1, рис. 1). Оператор *S* выполняется на уровне Sensor, *F* и *T* – на Edge, *G* – на Fog/Cloud, *V* – на Cloud. Такое распределение связывает логические этапы обработки данных с физическими вычислительными ресурсами.

Общая структура архитектуры и распределение операторов по уровням представлены на рисунке 1.

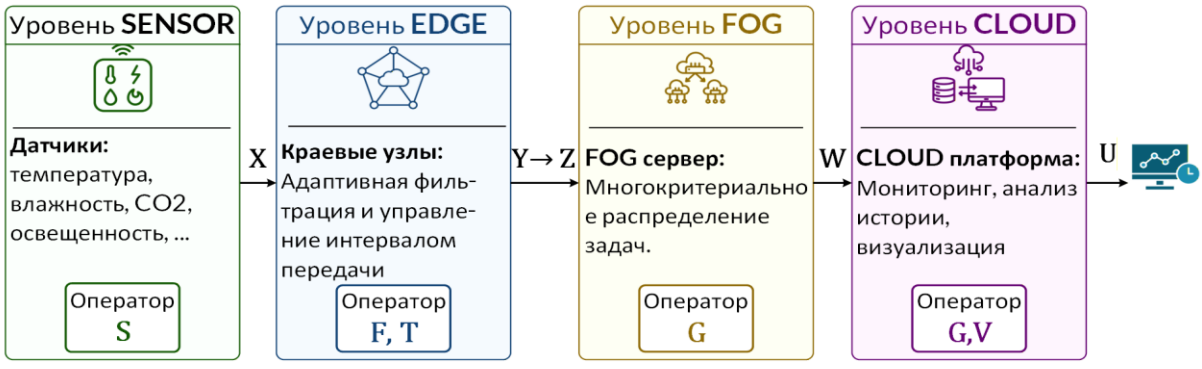



*Рисунок 1. Многоуровневая архитектура IoT, основанная на цепочке операторов*

Модульная структура программного комплекса и взаимодействие его компонентов представлены на UML-диаграмме компонентов (рис. 2). Операторы *S*, *F*, *T*, *G* и *V* реализованы как самостоятельные программные модули. Такое разбиение упрощает модификацию компонентов и масштабирование архитектуры.

(cc) BY

ISSN: 3030-3346

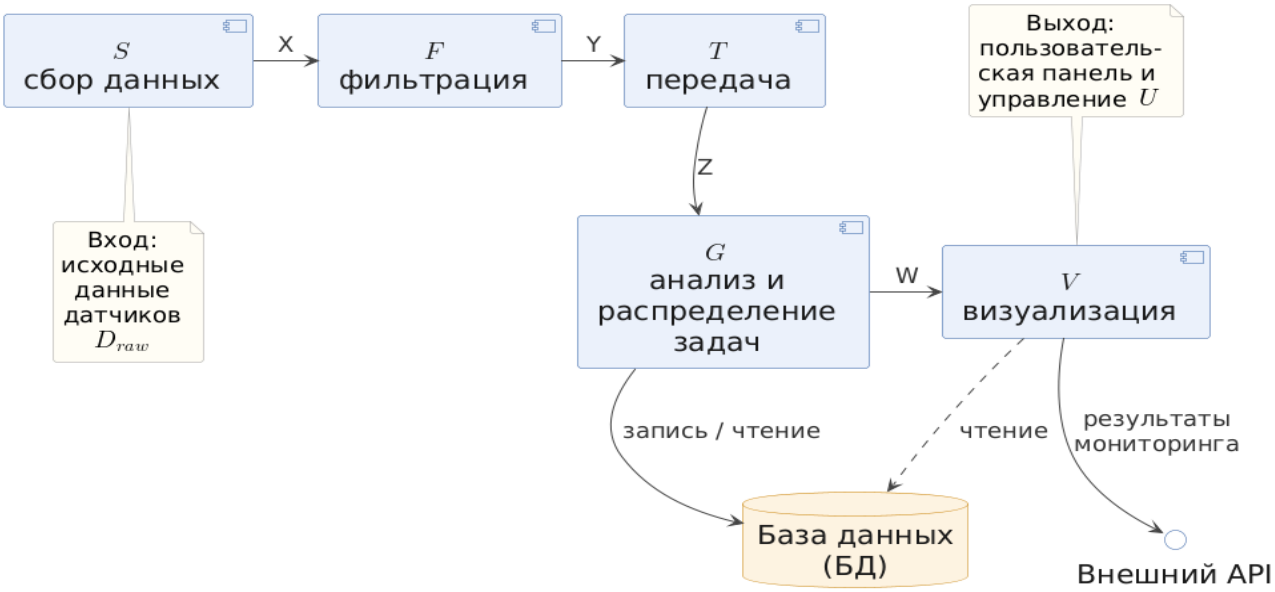



*Рисунок 2. Компонентная диаграмма программного комплекса (диаграмма построена в PlantUML на основе стандарта ISO/IEC 19505-2:2012)*

Последовательность выполнения операторов и взаимодействие между функциональными компонентами программного комплекса представлены на UML-диаграмме последовательности (рис. 3). Диаграмма отражает поэтапную обработку данных в цепочке $S \rightarrow F \rightarrow T \rightarrow G \rightarrow V$, включая сбор данных, фильтрацию, адаптивную передачу, анализ и визуализацию результатов в распределённой IoT-архитектуре.

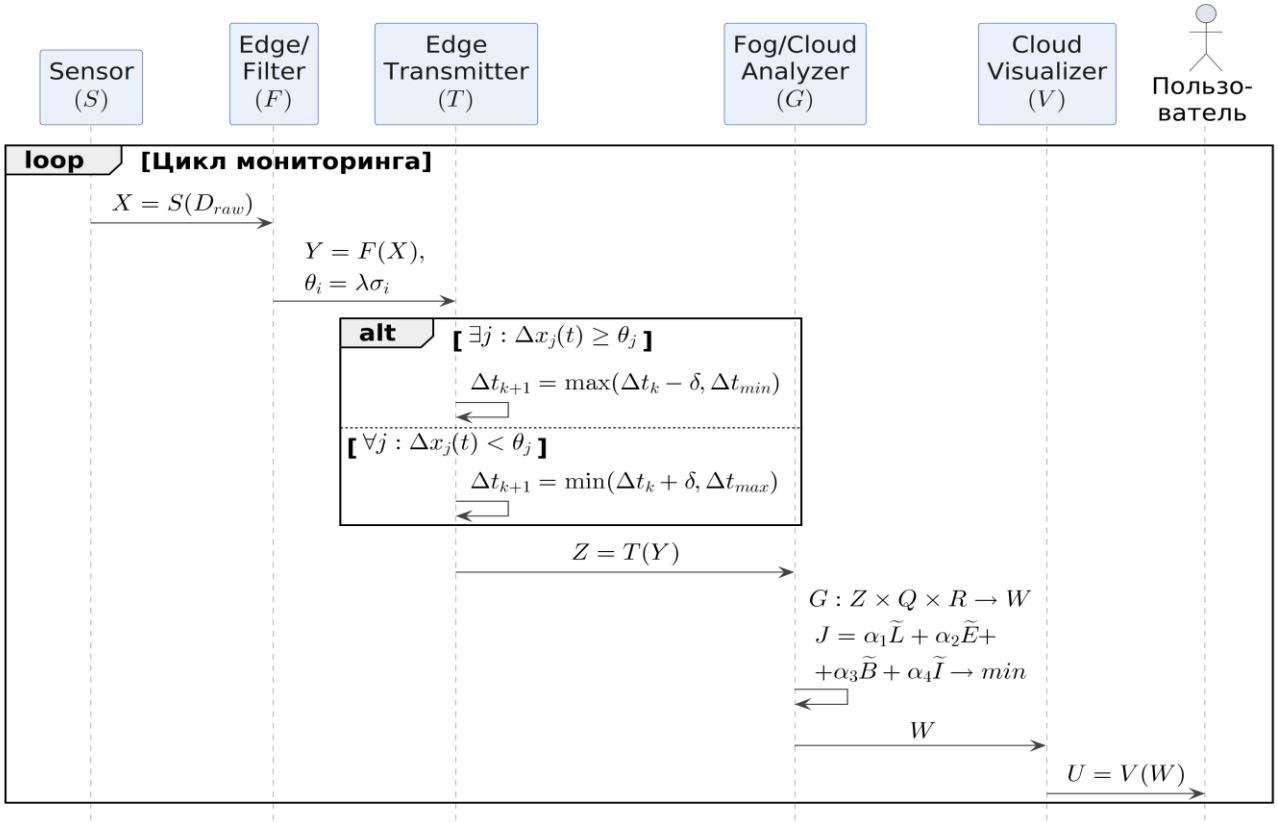



*Рисунок 3. UML-диаграмма последовательности выполнения цепочки операторов (диаграмма построена в PlantUML на основе стандарта ISO/IEC 19505-2:2012)*

(cc) BY 

### Экспериментальная оценка

Комплексная оценка эффективности предложенной архитектуры, основанной на цепочке операторов, была проведена на трех агропромышленных объектах Республики Каракалпакстан – двух теплицах и одном животноводческом комплексе. Период наблюдения для каждого объекта составлял не менее 30 дней. В эксперименте сравнивались два режима работы на одной и той же сенсорной инфраструктуре: централизованный режим управления и распределенный режим управления, основанный на цепочке операторов.

В режиме централизованного управления, принятом в качестве базового, данные датчиков передавались непосредственно на центральный вычислительный узел. Адаптивная фильтрация и управление интервалом передачи на уровне Edge не применялись.

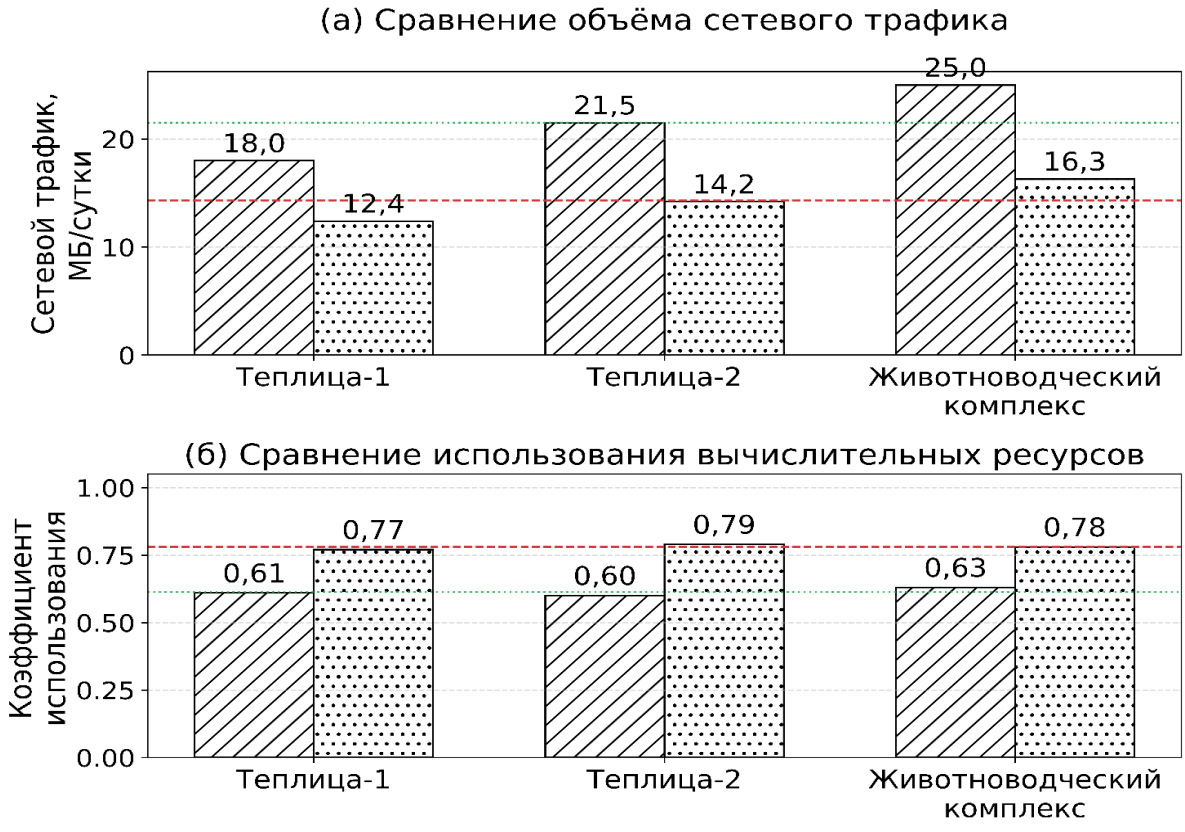
В предлагаемом режиме операторы *F* и *T* активировались на уровне Edge. Распределение вычислительных задач между ресурсами Fog и Cloud выполнялось оператором *G*. В обоих режимах состав датчиков, наблюдаемые параметры и среда эксперимента оставались неизменными.

Относительное уменьшение сетевого трафика рассчитывалось следующим образом:

$$R_D = \frac{D_{base} - D_{drop}}{D_{base}} \cdot 100\%$$

где $D_{base}$ и $D_{drop}$ представляют собой объем данных, передаваемых в режиме централизованного управления и предлагаемом режиме соответственно.

Эксперимент показал снижение сетевого трафика на 30–35%. Коэффициент использования вычислительных ресурсов увеличился с 0,61 до 0,78, а коэффициент сбалансированности нагрузки – с 0,72 до 0,94. Соответственно, дисбаланс *I=1-B* уменьшился с 0,28 до 0,06. Результаты подтверждают эффективность предварительной обработки на Edge и распределения задач между Fog и Cloud (рис. 4).




(cc) BY 

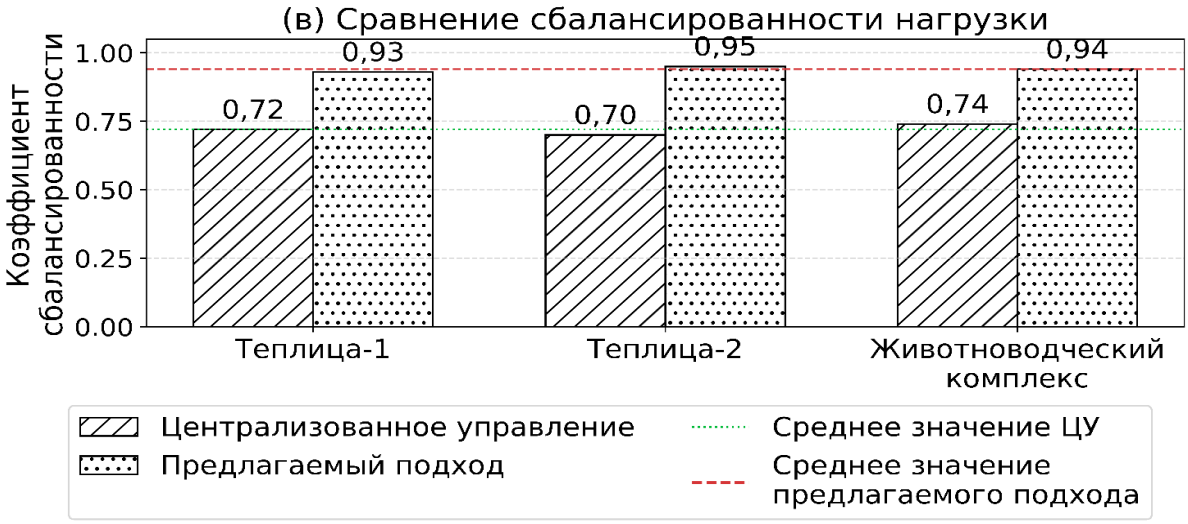



*Рисунок 4. Сравнение показателей централизованного и предлагаемого подходов на агропромышленных объектах*

**Обсуждение результатов**

Предложенная модель объединяет сбор, фильтрацию, адаптивную передачу, анализ, распределение задач и визуализацию в единой цепочке $S \to F \to T \to G \to V$. Композиция (6)–(7) задает согласованность информационного потока, а проекция (8)–(9) связывает операторы с уровнями Sensor–Edge–Fog–Cloud. В отличие от работ, где сокращение данных [11,12], планирование задач [13,14] и балансировка нагрузки [15] рассматриваются раздельно. Предложенный подход формализует их взаимодействие в общей архитектуре. Ограничением модели является фиксированный состав операторов, а автоматическая реконфигурация остается направлением дальнейших исследований.

**Заключение**

В работе разработана формальная модель, объединяющая сбор, фильтрацию, передачу, анализ и визуализацию данных в цепочке операторов $M = \langle S, F, T, G, V \rangle$. Определены функциональная композиция операторов и их проекция на уровни Sensor–Edge–Fog–Cloud. На этой основе синтезирована модульная многоуровневая программная архитектура.

Энергоэффективность адаптивного механизма передачи, реализованного оператором $T$, подтверждена результатами предыдущего исследования [6]. Экспериментальная оценка на архитектурном уровне в данной работе показала возможность снижения сетевого трафика на 30–35% по сравнению с централизованным подходом. Коэффициенты использования вычислительных ресурсов и сбалансированности нагрузки в предлагаемом режиме составили 0,78 и 0,94 соответственно.

Дальнейшие исследования будут включать автоматическую настройку состава операторов и параметров в соответствии с состоянием системы, а также адаптивную оптимизацию механизмов распределения задач.

**Список литературы**

1. S. Dhal et al. Internet of Things (IoT) in digital agriculture: An overview // Agronomy Journal. 2024. Vol. 116. P. 1144–1163.
2. M. S. Farooq, O. O. Sohail, A. Abid, S. Rasheed. A Survey on the Role of IoT in Agriculture for the Implementation of Smart Livestock Environment // IEEE Access. 2022. Vol. 10, P. 9483–9505, 2022.
3. B. Samandarov, A. Nishanov, O. Ruzibaev, D. Joldasbaev. Algorithmic maintenance of the feed ration formulation system for livestock farms // Second International Scientific and Practical Conference on Actual Problems of Mathematical Modeling and Information Technology:

APMMIT2024, Nukus, Uzbekistan, 2025, P. 040006.
4. Y. Kalyani, R. Collier. A Systematic Survey on the Role of Cloud, Fog, and Edge Computing Combination in Smart Agriculture // Sensors. 2021. Vol. 21, No. 17, P. 5922.
5. Жолдасбаев Д. Р. Задача балансировки нагрузки в системах основанных на интернета вещей (IoT) // “Актуальные и перспективные научные исследования”, Пенза, Россия, 2024, -С 34–36.
6. A. X.Nishanov, B. S.Samandarov, D. R.Joldasbayev, G. A.Gulmirzayeva. IoT qurilmalarda ma’lumot uzatish jarayonlarini adaptiv boshqarish algoritmi // Al-Farg‘oniy avlodlari. 2025. Vol. 1. No. 1. C. 100–105.
7. Sh. X. Tajibayev, D. R. Joldasbaev, G. B. Nietbaeva. Design of the Architecture of a Data Processing System in IoT Devices for Greenhouse Environments // Science and Education in Karakalpakstan. 2025. Vol. 2/2, No. 49, P. 219–223.
8. L. Kong et al. Edge-computing-driven Internet of Things: A Survey // ACM Computing Surveys. 2022. Vol. 55, No. 8. P. 1–41.
9. A.Hazra, P.Rana, M.Adhikari, T.Amgoth. Fog computing for next-generation Internet of Things: Fundamental, state-of-the-art and research challenges // Computer Science Review. 2023. Vol. 48. P. 100549.
10. M.Zolghadri, P.Asghari, S.E.Dashti, A.Hedayati. Resource allocation in Fog–Cloud Environments: State of the art // Journal of Network and Computer Applications. 2024. Vol. 227. P. 103891.
11. L. Pioli et al. An overview of data reduction solutions at the edge of IoT systems: a systematic mapping of the literature // Computing. 2022. Vol. 104, No. 8, P. 1867–1889.
12. F. M. Ribeiro Junior et al. Data reduction based on machine learning algorithms for fog computing in IoT smart agriculture // Biosystems Engineering. 2022. Vol. 223, P. 142–158.
13. P.Ntumba, N.Georgantas, V.Christophides Adaptive Scheduling of Continuous Operators for IoT Edge Analytics // Future Generation Computer Systems. 2024. Vol. 158. P. 277–293.
14. N. Kumari, A. Yadav, P. K. Jana. Task offloading in fog computing: A survey of algorithms and optimization techniques // Computer Networks. 2022. Vol. 214, P. 109137.
15. M.Ebrahim, A.Hafid. Resilience and load balancing in Fog networks: A Multi-Criteria Decision Analysis approach // Microprocessors and Microsystems. 2023. Vol. 101. P. 104893.
16. M.O.Solovev. Architecture and algorithms for using the internet of things in smart city systems // Automation. Modern Techologies. 2025. No.6. P. 272–274,
17. В. А. К. Камиль, Д. И. Мутин. Методы кластеризации сенсорных узлов на основе нечетких множеств // Информационные технологии моделирования и управления, 2024. 1(135). -С. 37–45.

(cc) BY 