DOI 10.51885/1561-4212_2025_4_212
ҒТАХР 20.53.17

БЛОКЧЕЙН ТЕХНОЛОГИЯСЫ НЕГІЗІНДЕ КОНТЕЙНЕРЛЕРДІ ЖОСПАРЛАУ

ПЛАНИРОВАНИЕ КОНТЕЙНЕРОВ НА ОСНОВЕ БЛОКЧЕЙН-ТЕХНОЛОГИЙ

BLOCKCHAIN-BASED CONTAINER SCHEDULING

Г.Ш. Сабырханова[1]*, О.З. Сембиев[1], Е.Р. Керимбеков[2], Б.С. Есмагамбетов[1]
[1]М. Әуезов атындағы Оңтүстік Қазақстан университеті, Шымкент қ., Қазақстан
[2]Ө. Жәнібеков атындағы Оңтүстік Қазақстан педагогикалық университеті, Шымкент қ., Қазақстан
*Жауапты автор: Сабырханова Гулзат Шалхарбаевна, e-mail: gulzat-077@mail.ru

Түйінді сөздер:
1-ші контейнер моделі,
2-ші сандық технология,
3-ші блокчейн,
4-ші Docker,
5-ші Kybernetes,
6-шы , Ethereum.

ТҮЙІНДЕМЕ
Бұл мақалада есептеу ресурстарын үлестіру процесінің ашықтығын, сенімділігін және автоматтандырылуын арттыру мақсатында блокчейн технологиясын пайдалана отырып, таратылған контейнерлік кластерлердегі тапсырмаларды жоспарлаудың инновациялық әдісі ұсынылады. Жүйе архитектурасы блокчейн желісінен, смарт-келісімшарттардан және контейнер оркестраторынан (мысалы, Kubernetes) тұрады, мұнда смарт-келісімшарттар жоспарлаушы ретінде әрекет етіп, шешімдерді блокчейнде тіркелген деректерге — тапсырма басымдылығы, ресурстық шектеулер мен орындалу тарихына сүйене отырып қабылдайды. Алгоритмдік негіз ретінде таратылған ортаға бейімделген және Solidity тілінде жүзеге асырылған модификацияланған басымдықты жоспарлау алгоритмі (Modified Priority Scheduling) пайдаланылды. Модельдеу арқылы алынған тәжірибелік нәтижелер ұсынылады, онда өнімділіктің негізгі метрикалары — кідіріс уақыты, тапсырманың орындалу уақыты және смарт-келісімшартты орындау шығындары — талданады. Алынған нәтижелер бұл әдістің қауіпсіздікке, қайталануға және тапсырмаларды автоматты басқаруға жоғары талаптар қойылатын жүйелерде — мысалы, білім беру платформаларында, децентрализденген есептеулерде және интеллектуалды таратылған жүйелерде — тиімді қолдануға болатынын көрсетеді.

Ключевые слова:
контейнерная модель 1,
цифровые технологии 2,
блокчейн 3, Docker 4,
Kybernetes 5, Ethereum 6.

АННОТАЦИЯ
В данной статье представлен инновационный подход к планированию задач в распределённых контейнерных кластерах с использованием блокчейн-технологии для повышения прозрачности, надёжности и автоматизации процесса распределения вычислительных ресурсов. Предложена архитектура системы, включающая в себя блокчейн-сеть, смарт-контракты и контейнерный оркестратор (например, Kubernetes), где смарт-контракты выполняют роль независимого планировщика, принимающего решения на основе зафиксированных в блокчейне данных о приоритетах, ресурсных ограничениях и истории выполнения. В качестве алго-

ритмической основы применён модифицированный приоритетный алгоритм планирования (Modified Priority Scheduling), адаптированный для условий распределённой среды и реализованный на языке Solidity. Представлены экспериментальные результаты моделирования, в которых анализируются ключевые метрики производительности — задержка, время выполнения задач, а также затраты на выполнение смарт-контракта. Полученные результаты демонстрируют потенциальную эффективность предложенного подхода для использования в системах с высокими требованиями к безопасности, воспроизводимости и автоматическому управлению задачами, таких как образовательные платформы, децентрализованные вычисления и интеллектуальные распределённые системы.

Keywords:
container model 1, digital technologies 2, blockchain 3, Docker 4, Kybernetes 5, Ethereum 6.

ABSTRACT

This paper presents an innovative approach to task scheduling in distributed container clusters using blockchain technology to enhance transparency, reliability, and automation in the allocation of computing resources. The proposed system architecture comprises a blockchain network, smart contracts, and a container orchestrator (e.g., Kubernetes), where smart contracts serve as independent schedulers, making decisions based on data recorded in the blockchain regarding task priorities, resource constraints, and execution history. A Modified Priority Scheduling algorithm, adapted for distributed environments and implemented in Solidity, is used as the algorithmic foundation. Experimental simulation results are presented, analyzing key performance metrics such as latency, task execution time, and the cost of smart contract execution. The obtained results demonstrate the potential effectiveness of the proposed approach for use in systems with high demands for security, reproducibility, and automated task management — such as educational platforms, decentralized computing environments, and intelligent distributed systems.

КІРІСПЕ

Таратылған есептеулер саласындағы заманауи үрдістер мен контейнерлік технологиялардың дамуы микросервис архитектурасының және Kubernetes пен Docker Swarm сияқты оркестраторлар арқылы басқарылатын масштабталатын платформалардың белсенді енгізілуіне ықпал етуде. Бұл құралдар контейнерлерді орналастыру, масштабтау және басқару міндеттерін тиімді шешеді, алайда олардың архитектурасы сенімді ортаға және орталықтандырылған басқаруға негізделген. Таратылған жүйелер жағдайында, әсіресе сенімсіз қатысушылардың немесе байланысы шектеулі түйіндердің болуы кезінде, тапсырмаларды жоспарлау үдерістерінің қауіпсіздігіне, ақауға төзімділігіне және ашықтығына қатысты бірқатар мәселелер туындайды.

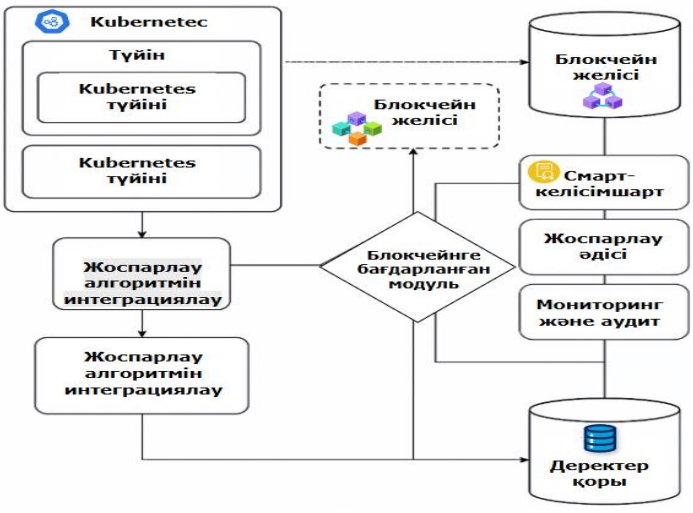
Блокчейн технологиясы децентрализация, жазбалардың өзгермейтіндігі және смарт-контрактілер арқылы әрекеттерді автоматтандыру сияқты қасиеттерге ие бола отырып, орталықтандырылған жоспарлаушылардың мәселелерін шешудің инновациялық тетігін ұсынады және есептеу ресурстарын басқарудың сенімдірек үлгілерін қалыптастырудың негізі бола алады (Altahat, Mohammad A. Dynamic, 2024).

Контейнерлік кластерлердегі тапсырмаларды жоспарлау орындалу басымдығынан бастап ағымдағы жүктемеге және қолжетімді ресурстарға дейінгі көптеген факторларды ескеруді талап етеді. Дәстүрлі жүйелерде мұндай шешімдерді орталықтандырылған жоспарлаушы қабылдайды, бұл бір нүктеден істен шығу қаупін, манипуляцияға бейімділікті және түйіндер арасындағы тиімсіз синхрондауды тудырады.

Блокчейнді пайдалану децентрализденген үлгіні жүзеге асыруға мүмкіндік береді, онда тапсырмалар туралы ақпарат, контейнерлердің күйі және орындалу нәтижелері барлық қатысушылар үшін қолжетімді қорғалған тізілімге жазылады. Мұндай үлгіде смарт-контрактілер жоспарлаушы функцияларын атқарып, ашықтықты, қайталанғыштықты және тіркелген деректер негізінде шешімдерді автоматты қабылдауды қамтамасыз ете алады. Бұл әдіс әсіресе географиялық тұрғыдан бөлінген кластерлерде, білім беру есептеу жүйелерінде және қатысушылар арасындағы сенімге қойылатын талаптары жоғары ортада өзекті болып табылады. Блокчейнге негізделген контейнер жоспарлау архитектурасының сипаттамалары 1-кестеде берілген. 1-суреттен жүйенің негізгі компоненттері мен олардың өзара әрекеттесу логикасын байқауға болады.

1-кесте. Ұсынылып отырған блокчейн-бағдарланған контейнер жоспарлау архитектурасының негізгі сипаттамаларын қамтиды

Сипаттама	Түсіндірме
Децентрализация	Шешім қабылдау блокчейн-желісіндегі смарт-контрактілер арқылы жүзеге асырылады.
Ашықтық	Жоспарлаушының барлық әрекеттері блокчейнге тіркеліп, тексеруге қолжетімді.
Қауіпсіздік және өзгермейтіндік	Тапсырмаларды тағайындау тарихы модификациядан қорғалған.
Kubernetes-пен интеграция	Архитектура қолданыстағы контейнер оркестраторларымен үйлесімді.
Икемді жоспарлау (MPS)	Modified Priority Scheduling алгоритмі жүктемеге бейімделіп қолданылады.
Кідірістерді бағалау және тіркеу	Жоспарлаудың барлық кезеңдері (растау, орындау) сақталып, талданады.
Орындалуды автоматтандыру	Смарт-контракт оператордың араласуынсыз тапсырмаларды бөлу үдерісін іске қосады.
Масштабталу мүмкіндігі	Көпкластерлі ортада жұмыс істеу қабілеті қамтамасыз етіледі.
Ескерту – автормен құрастырылған	



1-сурет. Kubernetes-кластеріндегі блокчейн-технологиялар негізіндегі контейнерлік жоспарлау архитектурасы

Ескерту – авторлармен (app.diagrams.net) негізінде құрастырылған

Суретте ұсынылған жүйенің архитектурасы берілген. Kubernetes-түйіндері тапсырмаларды орындаушы рөлін атқарады, сонымен қатар әрбір түйінде блокчейн-желімен әрекеттесетін кірістірілген модуль бар. Modified Priority Scheduling (MPS) алгоритмі смарт-контрактілерден алынған басымдықтар, ресурстар және орындалу тарихы туралы деректерді пайдаланады. Қабылданған шешімдер блокчейн-желісіндегі консенсус арқылы тексеріліп, расталады. Оқиғалар туралы ақпарат тіркеліп, кейінгі талдау үшін қолданылады.

Схема сонымен қатар Kubernetes-түйіндердің, блокчейн-желінің, смарт-контрактілердің және жоспарлау алгоритмдерінің өзара әрекеттесуін көрсетеді. Контейнерлік тапсырмалар қауіпсіздік саясаты, басымдықтар және ресурстардың қолжетімділігі ескеріліп бөлінеді, ал қабылданған шешімдер туралы деректер блокчейнге жазылады. Жоспарлау алгоритмінің интеграциясы мен мониторинг on-chain және off-chain деңгейлерінде жүзеге асырылады.

ЗЕРТТЕУ МАТЕРИАЛДАРЫ МЕН ӘДІСТЕРІ

Ұсынылып отырған әдіс контейнерлік кластерлердегі тапсырмаларды жоспарлау механизмдерімен блокчейн-технологияларын біріктіру арқылы орталықтандырылған әдіс-тердің кемшіліктерін жоюға бағытталған. Әдістің негізінде смарт-контрактілер жатыр — олар блокчейн-желісінде орындалатын, есептеу түйіндеріне тапсырмалар тағайындау логикасын іске асыратын автономды бағдарламалар.

Жоспарлаушының барлық әрекеттері (тапсырмаларды тіркеу, қолжетімді ресурстарды тексеру, басымдықтарды есептеу және орналастыру шешімін қабылдау) блокчейнге тіркеліп, олардың өзгермейтіндігі мен барлық қатысушылар үшін ашықтығы қамтамасыз етіледі (Zhou, Ruiting, Zongpeng Li, and Chuan Wu). Түйіндердің ағымдағы жүктемесі, орындалу тарихы және тапсырмалар мәртебелері транзакциялар түрінде жиналып, нақты уақыт режимінде талдауға қолжетімді болады.

Әдістің алгоритмдік өзегі – Modified Priority Scheduling (MPS) модификацияланған басымдықты жоспарлау алгоритмі, ол шектеулі ресурстар мен жоғары динамика жағдайына бейімделген. Дәстүрлі нұсқалардан айырмашылығы, бұл алгоритм қосымша параметрлерді — орындалу кідірісін, түйіннің беделін және өңдеу шығындарын ескеріп, тапсырмаларды неғұрлым әділ және дәл бөлуді қамтамасыз етеді. MPS алгоритмін іске асыратын смарт-контракт Solidity тілінде жазылып, Ethereum-үйлесімді ортада орындалады, бұл шешімдердің қайта өндірілуін және сыртқы араласудан қорғалуын қамтамасыз етеді.

Әдіс permissioned-блокчейн – желіні қолдануды көздейді, оның құрамына таратылған кластердің барлық есептеу түйіндері кіреді. Әрбір түйінге блокчейн-клиент сәйкес келеді, ол консенсус механизміне қатысып, тапсырмалар, ресурстар мен мәртебелер туралы деректерді сақтайды, сондай-ақ жоспарлау шешімдерін қабылдау үшін смарт-контрактілерді орындайды.

Қауіпсіздік пен архитектураның икемділігіне қойылатын талаптарға байланысты блокчейн платформасы ретінде Hyperledger Fabric (корпоративтік қорғалған орта үшін) немесе Ethereum (кеңейтілетін және бағдарламаланатын логика қажет болған жағдайда) таңдалуы мүмкін. Барлық түйіндер бір мезгілде Kubernetes-тегі жұмысшы түйіндер (workers) және блокчейн-желідегі валидаторлар ретінде жұмыс істейді, бұл контейнерлік оркестрация мен децентрализденген жоспарлау арасындағы тығыз интеграцияны қамтамасыз етеді (Lei, Yu, and S. Yu Philip, 2020).

Тиімді деректер алмасуын қамтамасыз ету үшін тапсырмаларға қойылатын талаптар, ресурстар сипаттамалары және мәртебелер тікелей блокчейнге жазылады. Ал ауыр артефактілер – контейнер бейнелері мен журнал файлдары – off-chain деңгейінде

сақталады, олардың криптографиялық хэшттері блокчейнде тіркеліп, тексерілуді қамтамасыз етеді. Смарт-контрактілер жүйеде негізгі рөл атқарады: олар жоспарлау, ресурстарды орналастыру, қолжетімділікті тексеру, желілік кідірістер, энергия тиімділігі және қауіпсіздік саясатын сақтау ережелерін анықтайды.

Осы модельдің орталық элементі — Modified Priority Scheduling (MPS) алгоритмі, смарт-контракт түрінде жүзеге асырылып, жүйенің параметрлерін динамикалық түрде бағалайды және блокчейнде бекітілген шарттар негізінде шешім қабылдайды. Әрбір шешім желідегі басқа қатысушылармен Practical Byzantine Fault Tolerance (pBFT) консенсус механизмі арқылы тексеріледі, бұл тапсырмаларды бұрмалау немесе қате тағайындау мүмкіндігін болдырмайды.

Жүзеге асыру кезеңдері мыналарды қамтиды:

1. Kubernetes-кластерін және блокчейн-желісін орналастыру;
2. Смарт-контрактілерді құру және жүктеу;
3. Жоспарлаушы логикасын off-chain-қосымша ретінде жүзеге асыру;
4. Блокчейнмен өзара әрекеттесу және мониторинг жүйесімен интеграция.

Қосымша, бұл әдіс алгоритмді жүктеменің өзгеруіне бейімдеуді және болжамдық талдау үшін машиналық оқыту элементтерін енгізуді қарастырады. Ұсынылып отырған әдіс дәстүрлі шешімдерден ерекшеленіп, смарт-контрактілер арқылы жоғары ашықтықты, ақауға төзімділікті және сенімділікті қамтамасыз етеді. Бұл әдіс масштабталатын, географиялық тұрғыдан бөлінген немесе сенімсіз ортада қолдануға қолайлы және Kubernetes сияқты қолданыстағы контейнер басқару жүйелерімен интеграциялана алады.

НӘТИЖЕЛЕР ЖӘНЕ ОЛАРДЫ ТАЛҚЫЛАУ

Эксперимент аясында Modified Priority Scheduling (MPS) алгоритмі смарт-контракт ішінде жүзеге асырылып, контейнерлік ортада (Docker) орналастырылды. Зерттеудің мақсаты – гибридті архитектураны (PHP-платформа, Go арқылы интеграция, контейнерде Ethereum-смарт-контракт) пайдалану кезінде тапсырмалардың орындалу уақыты мен ресурстық сипаттамаларын талдау. Контейнер мен блокчейн негізіндегі модельдердің орындау уақыты шартты түрде салыстырылып көрсетілген. Контейнерлік ортада CPU мен жадтың жүктемесі шамамен тұрақты болғанымен, смарт-келісім енгізілген жағдайда шағын кідіріс байқалды (0.07-0.09 секунд шамасында). Бұл айырмашылық модельдің қауіпсіздік тексеру процесімен түсіндіріледі. Бұл деректер CPU пайдалану пайызының тұрақтылығын (шамамен 12-13 %), орындалу уақытының 0.21-0.27 с диапазонында екендігін және жадты тұтынудың 190-210 КБ аралығында сақталатынын көрсетеді.

Алгоритм: TestResultContract_Execution_Analysis
Кіріс: testId, userId, result мәндері
Шығыс: blockchain transaction hash және орындау метрикалары
1. Смарт-контракт құрылымы:
a) struct TestResult → {testId, userId, result}
b) mapping(testId → TestResult)
c) event ResultStored(testId, userId, result)
2. Нәтижені сақтау:
Функция storeResult(userId, testId, result)
– Егер testId бұрын сақталмаған болса:
testResults[testId] ← {userId, testId, result}
emit ResultStored(testId, userId, result)
– Әйтпесе, require → “Result already stored for this testId”
3. Нәтижені алу:

```
Функция getResult(testId)
    res ← testResults[testId]
    return (res.userId, res.testId, res.result)
```

4. Эксперимент кезінде:
– Тест нәтижесі енгізіледі (мысалы, 85 ұпай)
– Smart-contract орындалады және transaction hash алынады
– Пайдаланушыға метрикалар тіркеледі:
- cpu_sys_seconds
- cpu_usage_percent
- execution_time_seconds
- mem_alloc_bytes
- smart_contract_delay_time
- smart_contract_execution_time
- transaction_hash

5. Нәтижелер JSON форматында сақталады:

```
{
  "cpu_usage_percent": "12.5",
  "execution_time_seconds": "0.210",
  "mem_alloc_bytes": 193324,
  "smart_contract_execution_time": "0.273",
  "transaction_hash": "0xb79e2c6f9...",
  "user_id": "21"
}
```

6. Талдау кезеңі:
- Әр транзакция бойынша орташа CPU, уақыт, жад және delay есептеледі
- Smart-contract тиімділігі бағаланады.

Әрбір контейнерде орналастырылған Modified Priority Scheduling (MPS) алгоритміне негізделген смарт-контракт ішінде орындалатын тапсырма үшін негізгі есептеу және ресурстық параметрлерді қамтитын математикалық модель құрастырылды.

Жалпы процессорлық уақыт CPU_i контейнер ішіндегі жүйелік уақыт пен пайдаланушы уақыттарының қосындысы ретінде есептеледі:

$$CPU_i = CPU_{SYS} + CPU_{USER} \quad (1)$$

мұнда: CPU_i −жалпы процессорлық уақыт; CPU_{SYS} − жүйелік режим уақыты; CPU_{USER} − пайдаланушы режим уақыты.

Смарт-контракт логикасын орындау уақыты SCD_i деп белгіленеді және Ethereum виртуалды машинасында орындалу кезінде алынатын smart_contract_execution_time параметрімен өлшенеді. Тапсырманың жалпы нақты орындалу уақыты ET_i execution_time_seconds параметріне сәйкес келеді.

Күту уақыты WT_i толық орындалу уақыты мен CPU-ды белсенді пайдалану және смарт-контракт орындалуының қосындысы арасындағы айырмашылық ретінде есептеледі:

$$WT_i = ET_i - (CPU_i + SCD_i) \quad (2)$$

мұнда: WT_i − күту уақыты (блокталу/кідіріс); ET_i − тапсырманың нақты толық орындалу уақыты; SCD_i) − смарт-контракт логикасының орындалу уақыты (EVM).

Бұл көрсеткіш ықтимал блоктауларды немесе кідірістерді бағалауға мүмкіндік береді. Егер барлық тапсырмалар бір мезгілде берілсе (AT_i=0), онда Turnaround Time (TAT_i) мәні ET_i-ге теңестіріледі.

Ақырында, жадты пайдалану MU_i mem_alloc_bytes параметрін мегабайтқа түрлендіру арқылы анықталады:

$$MU_i = \frac{mem\_alloc\_bytes}{1024^2} \quad (3)$$

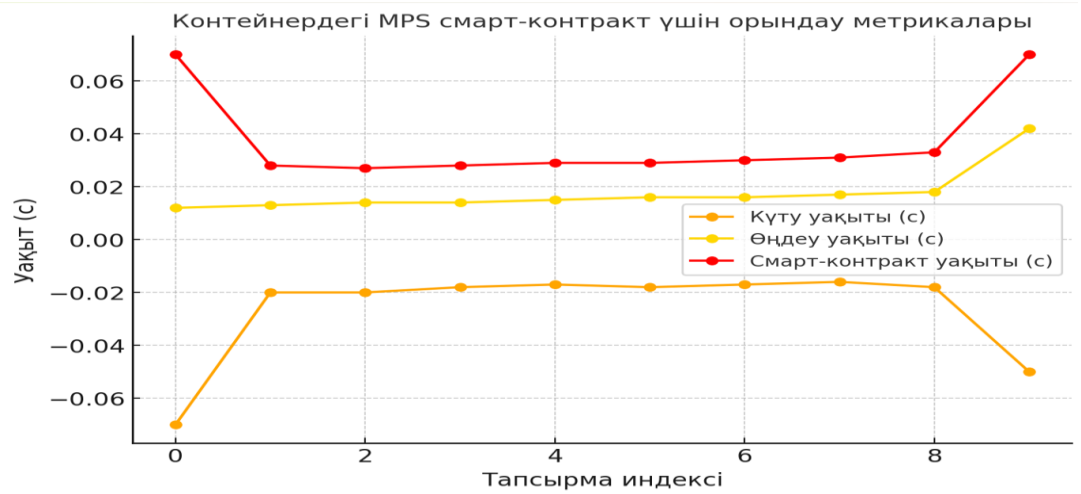
мұнда: MU_i – жадты пайдалану.

Осы модель әрбір тапсырманың таратылған ортадағы жүріс-тұрысын сандық сипаттауға мүмкіндік береді және контейнерленген блокчейн-қосымшалардың өнімділігін талдау негізі болып табылады.

Контейнерлік ортада орындалған MPS-смарт-контракт алгоритмінің тиімділігі 2-кестеде берілген метрикалар негізінде бағаланды. Кестеден байқалғандай, орташа орындау уақыты 0,07 с шегінде қалыптасып, CPU пайдалану тұрақты деңгейде сақталған. Бұл үрдіс 2-суретте көрсетілгендей, күту уақытының азаюымен және смарт-контракттың орындалу тұрақтылығымен сипатталады. Ал 3-суретте тапсырмалар бойынша жадты пайдалану нәтижелері көрсетіліп, контейнерлік ортаның ресурстарды теңгерімді бөлу қабілеті айқындалған.

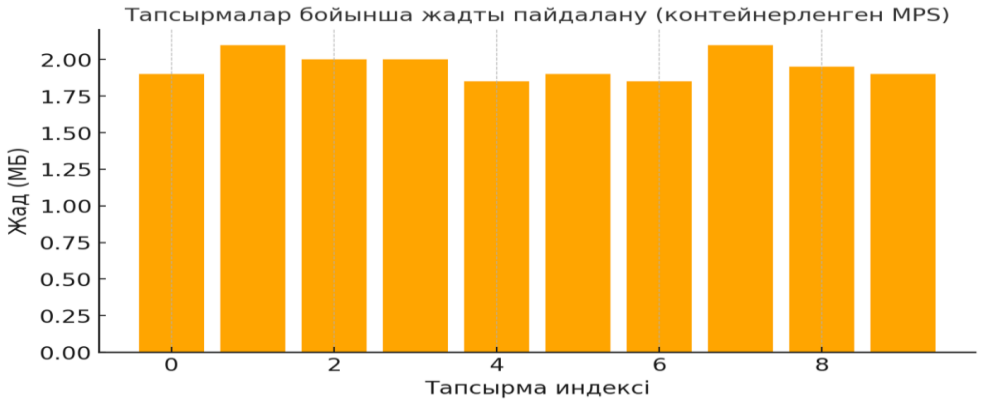
2-кесте. Контейнердегі MPS-смарт-контракт үшін орындалу және жадты пайдалану метрикалары

CPU жүйелік уақыты (с)	CPU пайдаланушы уақыты (с)	CPU жалпы уақыты (с)	Смарт-келісімшартты орындау уақыты (с)	Жалпы орындалу уақыты (с)	Күту уақыты (с)	Өңдеу уақыты (ТАТ) (с)	Жадты пайдалану (МБ)
0,003200	0,007800	0,011000	0,072546	0,012080	-0,071466	0,012080	1,843758
0,002800	0,005700	0,008500	0,029266	0,014281	-0,023485	0,014281	2,083267
0,003000	0,006700	0,009700	0,027026	0,014718	-0,022008	0,014718	2,001610
0,003100	0,006900	0,010000	0,029266	0,014966	-0,024300	0,014966	2,018852
0,002700	0,006500	0,009200	0,029266	0,015191	-0,023275	0,015191	1,829720
0,003300	0,007000	0,010300	0,027026	0,015554	-0,021772	0,015554	1,925240
0,003500	0,007300	0,010800	0,029266	0,016903	-0,023163	0,016903	1,844055
0,002900	0,006200	0,009100	0,029266	0,016944	-0,021422	0,016944	2,105782
0,003600	0,007600	0,011200	0,029266	0,018614	-0,021852	0,018614	1,936638
0,004500	0,009500	0,014000	0,072546	0,042865	-0,043681	0,042865	1,928688
Ескерту – деректер негізінде автормен құрастырылған							



2-сурет. Контейнердегі MPS смарт-контрактының орындалу метрикалары

Ескерту – автормен құрастырылған



3-сурет. Тапсырмалар бойынша жадты пайдалану

Ескерту – автормен құрастырылған

Бұл зерттеуде контейнерлік ортада орындалатын смарт-контракт ішінде жүзеге асырылған Modified Priority Scheduling (MPS) алгоритміне негізделген интеграцияланған жоспарлау моделі ұсынылды. Эксперименттік деректер негізгі метрикаларды қамтиды: жалпы процессорлық уақыт (CPU Total), смарт-контракт логикасын орындау уақыты (Smart Contract Time), тапсырманың жалпы орындалу уақыты (Execution Time), күту уақыты (Waiting Time), толық өңдеу уақыты (Turnaround Time) және мегабайттағы жадты пайдалану (Memory Usage).

Нәтижелер көрсеткендей:

1. Орташа күту уақыты – 0,0296 с, бұл блокчейн-компонент пен CPU-нұсқаулардың параллель орындалуын, блоктаушы кідірістердің жоқтығын көрсетеді;
2. Тапсырманың орташа толық орындалу уақыты (Turnaround Time) 0,0182 с, ал смарт-контракт логикасын орындау уақыты есептеу уақытынан асып түсіп, 0,0375 с құрайды, бұл Ethereum виртуалды машинасында транзакцияларды өңдеудің күрделілігімен түсіндіріледі;
3. Жадты тұтыну тұрақты болып, орташа 1.95 МБ деңгейінде сақталды, бұл ұсынылған архитектураны бұлтты және edge-ортада масштабталатын орналастыруға қолайлы етеді.

Осылайша, смарт-контракт ішінде контейнерленген MPS-алгоритмді орындау өңдеудің жоғары тиімділігін, жүйе жүріс-тұрысының болжамдылығын және ресурстарды оқшаулауды төмен кідіріспен қамтамасыз етеді. Бұл әсіресе уақытқа сындарлы таратылған қосымшалар үшін маңызды.

Модельдің ерекшелігі — MPS алгоритмінің смарт-контракт ішінде жүзеге асырылуы:

1. Тапсырмалардың «ашығу» мәселесін болдырмайды;
2. Асинхронды әрі тиімді орындалуды қамтамасыз етеді;
3. Контейнерлік оқшаулау арқылы ресурстардың тұтынуын бақылауға мүмкіндік береді.

Алынған нәтижелер Go – PHP – Solidity интеграциясы бар контейнерленген блокчейн-жүйесі төмен кідіріс, ресурстардың тұрақты тұтынуы және жоспарлаушының болжамды жұмыс істеуін қамтамасыз ететінін дәлелдейді.

Блокчейнге негізделген жоспарлау әдісі таратылған есептеу ортасында ашықтықты, қайта өндірілуін және қатысушылар арасындағы сенімділікті арттырады. Смарт-контрактілер автономды жоспарлаушылар ретінде шешім қабылдау процесін автоматтандырып қана қоймай, оны тексерілетін және сыртқы араласудан қорғалған етеді. Бұл әсіресе сенімсіз немесе географиялық тұрғыдан қашық қатысушылар бар жағдайда маңызды.

Алайда, тәжірибелік қолдануда бірқатар техникалық шектеулер байқалады:

1. блокчейнге жүктеме жоғарылаған кезде транзакцияларды растау уақыты артып, тапсырмаларды жоспарлау мен бөлуге кідірістер енгізеді;
2. смарт-контракт ішіндегі есептеулер газ шығындары мен орындалатын логиканың күрделілігі бойынша шектеледі, сондықтан кодты оңтайландыру және есептеулердің бір бөлігін off-chain компоненттерге көшіру қажет.

Дегенмен, ұсынылған модель манипуляцияларға тұрақты, істен шығуға төзімді және бар ресурстарды басқару жүйелерімен икемді интеграцияға мүмкіндік береді. Болашақта консенсус протоколдарын жетілдіру, өнімділігі жоғары блокчейн-платформаларды енгізу және гибридті on-chain/off-chain шешімдерді пайдалану бұл технологияның масштабталуы мен қолданылуын айтарлықтай арттыра алады.

ҚОРЫТЫНДЫ

Ұсынылған блокчейн-бағдарланған контейнерлік жоспарлау архитектурасы таратылған есептеу ортасында тапсырмаларды децентрализованған басқару үшін жоғары тиімділік пен өзектілікті көрсетеді. Kubernetes сияқты контейнерлік платформалармен интеграция және смарт-контракт түрінде жүзеге асырылған модификацияланған басымдықты жоспарлау алгоритмін (MPS) қолдану шешім қабылдау процесінде автоматтандырудың, икемділіктің және қорғалғандықтың жоғары деңгейіне қол жеткізуге мүмкіндік береді.

Блокчейнді сенімді дереккөзі ретінде пайдалану орталықтандырылған сенім қажеттілігін жояды және архитектураның істен шығуға және сыртқы әсерлерге төзімділігін арттырады. Эксперименттік нәтижелер түрлі басымдықтағы тапсырмалар үшін осы әдістің қолданылуын дәлелдеді, тұрақтылықты және ашықтықты қамтамасыз ете отырып, қақтығыстар мен тоқтап қалуларды азайтты. Шектеулерге қарамастан (транзакцияларды растау кідірістері мен смарт-контракт орындау шығындары), ұсынылған модель қазіргі таңда-ақ таратылған кластерлерде тиімді қолданыла алады.

Болашақта жүктемені болжау, тапсырмалардың жүріс-тұрыс үлгілерін талдау және жоспарлау алгоритмін нақты уақыт режимінде бейімдеу үшін машиналық оқыту механизмдерін енгізу көзделуде. Бұл басқарудың дәлдігін, тұрақтылығын және интеллектуалдылығын одан әрі арттырады.

ӘДЕБИЕТТЕР ТІЗІМІ

Altahat, M.A. (2024). Dynamic management of virtual machine and container scheduling in multi-cloud data centers (Doctoral dissertation, Concordia University). Concordia University. https://spectrum.library.concordia.ca/id/eprint/994071/1/Altahat_PhD_F2024.pdf

Chen, H., Chen, W., Zeng, G., & Li, K. (2024). Container scheduling algorithms for distributed cloud environments. Processes, 12(9), 1804. https://doi.org/10.3390/pr12091804

Lei, Y., & Philip, S. Y. (2020). Container scheduling in blockchain-based cloud service platform. In 2020 IEEE International Conference on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom) (pp. 1–8). IEEE. https://doi.org/10.1109/ISPA-BDCloud-SocialCom-SustainCom51426.2020.00148

Choi, B. C., Kim, J., Lee, J., & Lee, Y. (2012). Container scheduling: Complexity and algorithms. Production and Operations Management, 21(1), 115–128. https://doi.org/10.1111/j.1937-5956.2011.01238.x

Zhang, W., Wang, C., Tang, F., & Xu, C. (2018). Cost-efficient and latency-aware workflow scheduling policy for container-based systems. In 2018 IEEE 24th International Conference

on Parallel and Distributed Systems (ICPADS) (pp. 1–8). IEEE. https://doi.org/10.1109/PADSW.2018.8644945

Zhou, R., Li, Z., & Wu, C. (2018). Scheduling frameworks for cloud container services. IEEE/ACM Transactions on Networking, 26(1), 436–450. https://doi.org/10.1109/TNET.2017.2781200

Chen, F., Zhou, X., & Shi, C. (2019). The container scheduling method based on the min-min in edge computing. In Proceedings of the 4th International Conference on Big Data and Computing (pp. 1–6). ACM. https://doi.org/10.1145/3335484.3335506

Luo, X., & Li, P. (2022). Learning-based off-chain transaction scheduling in prioritized payment channel networks. IEEE Journal on Selected Areas in Communications, 40(12), 3589–3599. https://doi.org/10.1109/JSAC.2022.3213333

Zou, A., Wang, H., Chen, L., & Zhou, J. (2025). A survey of real-time scheduling on accelerator-based heterogeneous architecture for time critical applications. arXiv preprint arXiv:2505.11970. https://doi.org/10.48550/arXiv.2505.11970

GeeksforGeeks. (n.d.). Preemptive priority CPU scheduling algorithm. GeeksforGeeks. https://www.geeksforgeeks.org/preemptive-priority-cpu-scheduling-algortithm/

Zheng, Z., Xie, S., Dai, H. N., Chen, X., & Wang, H. (2017). An overview of blockchain technology: Architecture, consensus, and future trends. In 2017 IEEE International Congress on Big Data (BigData Congress) (pp. 557–564). IEEE. https://doi.org/10.1109/BigDataCongress.2017.85

Kim, T., & Kim, H. J. (2020). Blockchain-based service performance evaluation method using native cloud environment. In 2020 International Conference on Software Security and Assurance (ICSSA) (pp. 1–6). IEEE Computer Society. https://doi.org/10.1109/ICSSA51225.2020.00009

Bakayov, V., & Custură, A. (2020). Blockchain evolution (Tech. Rep.). Research Institute, Amsterdam, Netherlands.

López-Pimentel, J. C., Rojas, O., & Monroy, R. (2020). Blockchain and off-chain: A solution for audit issues in supply chain systems. In Proceedings of the IEEE International Conference on Blockchain (Blockchain) (pp. 126–133). IEEE. https://doi.org/10.1109/Blockchain50366.2020.00022

Sultan, N. A., & Qasha, R. P. (2023). Container-based virtualization for blockchain technology: A survey. Jordanian Journal of Computers & Information Technology, 9(3), 233–245.

Авторлар туралы мәліметтер
Информация об авторах
Information about authors

Сабырханова Гулзат Шалхарбаевна – автоматтандыру, телекоммуникация және басқару кафедрасының докторанты, М. Әуезов атындағы Оңтүстік Қазақстан университеті, Шымкент қ., Қазақстан

Сабырханова Гулзат Шалхарбаевна – Докторант кафедры автоматизации, телекоммуникаций и управления, Южно-Казахстанский университет имени М. Ауэзова, г. Шымкент, Казахстан

Sabyrkhanova Gulzat Shalkharbaevna – Doctoral Student, Department of Automation, Telecommunications and Control, M. Auezov South Kazakhstan University, Shymkent, Kazakhstan,
e-mail: gulzat-077@mail.ru,
ORCID: https://orcid.org/0009-0006-6868-2185

Сембиев Ордабай Зайтаевич – техника ғылымдарының докторы, профессор, М. Әуезов атындағы Оңтүстік Қазақстан университеті, Шымкент қ., Қазақстан
Сембиев Ордабай Зайтаевич – доктор технических наук, профессор, Южно-Казахстанский университет имени М. Ауэзова, г. Шымкент, Казахстан
Sembiyev Ordabay Zaitayevich – Doctor of Technical Sciences, professor, M. Auezov South Kazakhstan University, Shymkent, Kazakhstan,
e-mail: ordabai@mail.ru,
ORCID: https://orcid.org/0000-0002-3961-0266

Керімбеков Ержан Рахымжанұлы – PhD, Ө. Жәнібеков атындағы Оңтүстік Қазақстан педагогикалық университеті, Шымкент қ., Қазақстан
Керимбеков Ержан Рахимжанович – PhD, Южно-Казахстанского педагогического университета им. У. Жанибекова, г. Шымкент, Казахстан
Kerimbekov Yerzhan Rakhimzhanuly – doctor PhD, South Kazakhstan State Pedagogical University, Shymkent, Kazakhstan,
e-mail: kerimbekov.yerzhan@okmpu.kz,
ORCID: https://orcid.org/0000-0002-6116-3669,

Есмагамбетов Булат-Батыр Саухымович – техника ғылымдарының докторы, профессор, М. Әуезов атындағы Оңтүстік Қазақстан университеті, Шымкент қ., Қазақстан
Есмагамбетов Булат-Батыр Саухымович – доктор технических наук, профессор, Южно-Казахстанский университет имени М. Ауэзова, г. Шымкент, Казахстан
Bolat-Batyr Esmagambetov – Doctor of Technical Sciences, professor, M. Auezov South Kazakhstan University, Shymkent, Kazakhstan,
e-mail: bulatbatyr@mail.ru,
ORCID: https://orcid.org/0000-0003-2825-5200