УДК 681 /324
DOI https://doi.org/10.32782/tnv-tech.2024.6.13

ОБГРУНТУВАННЯ ВИБОРУ ДИСЦИПЛІНИ ОБСЛУГОВУВАННЯ ЗАЯВОК У РОЗПОДІЛЕНИХ СИСТЕМАХ ОБРОБКИ ІНФОРМАЦІЇ

Сімоненко А. В. *– старший викладач кафедри обчислювальної техніки Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського»*
ORCID ID: 0000-0003-4056-1753

У статті розглядаються підходи та найпоширеніші дисципліни обслуговування заявок (алгоритми вибору заявки для виділення часу на процесорі. Дано класифікацію алгоритмів планування. Виділено пріоритетні та безпріоритетні дисципліни. Представлені алгоритми обслуговування як з одною чергою , так и кількома чергами. Показано переваги та недоліки представлених дисциплін, а також особливості їх застосування. З представленого аналізу можна зробити висновки, що всі описані алгоритми виконують планування у часі так как ориентировані на обчислювальну середу с одним прцесором. Однак необхідно враховувати, що сучасні обчислювальні системи обробки інформації мають характеристики притаманні розподіленим системам. Особливість сучасних систем полягає в тому, що безлічі заявок середовище виконання (обчислювальні ресурси) може надавати безліч ресурсів одночасно. В них як обчислювальні ресурси, так і завдання розподілені у просторі. Таки особливості характерні для глобальних GRID та CLOUD систем обробки інформації. Крім цього завдання та ресурси можуть мати характеристики однородности и неоднорідності. За таких характеристик дисципліни, призначені для одно процесорних систем, не можуть бути використані для розподілених систем обробки інформації. У таких системах планувальник має інформацію про безліч заявок, що вимагають обслуговування, як і у відомих системах обслуговування та інформацію про безліч ресурсів, які мають можливість обслуговувати заявки. До цього планувальник має інформацію про вимоги кожної заявки до передбачуваного ресурсу. Крім того, планувальник має інформацію про характеристики кожного ресурсу.

Така інформація про заявки та ресурси може дозволити планувальнику забезпечити оптимальне рюрозподілення заявок на ресурси, враховуючи особливості вимог заявок та характеристик ресурсів. Таким чином у розподіленій системі обробки інформації потрібен новий підхід до проектування нових типів планувальників – тобто. просторових планувальників. У статті подано теоретичне обґрунтування нового підходу для вибору та практики просторового планувальника. У статті показано, що відносини заявка-ресурс у розподілених системах можна уявити у вигляді дводольного графа. А задача розподілу заявки на ресурс зводиться до задачі пошуку максимального паросполучення у зваженому чи незваженому дводольному графі.

Ключові слова: *планування, диспетчеризація, складання розкладу, розподілені системи.*

Simonenko A. V. Justification for choosing a discipline for servicing requests in distributed information processing systems

The article examines approaches and the most common application service disciplines (algorithms for application selection for allocating time on the processor. The classification of scheduling algorithms is given. Priority and non-priority disciplines are distinguished. Service algorithms with both one queue and several queues are presented. The advantages and disadvantages of the presented disciplines, as well as the peculiarities of their application, are shown. From the presented analysis, it can be concluded that all the described algorithms perform planning in time, as they are focused on a computing environment with one processor.However, it is necessary to take into account that modern computing systems of information processing have characteristics inherent in distributed systems. The peculiarity of modern systems is that the execution environment (computing resources) can provide many resources simultaneously for many applications. In them, both computing resources and tasks are distributed in space. Such features are characteristic of global GRID and CLOUD information processing systems. In addition, tasks and resources can have characteristics of homogeneity and heterogeneity. With such characteristics, the disciplines intended for single-processor systems cannot be used for distributed information processing systems. In such systems, the scheduler has information about

the number of requests requiring service, as in known service systems, and information about the number of resources that have the ability to service the requests. Before this, the scheduler has information about the requirements of each application for the intended resource. In addition, the scheduler has information about the characteristics of each resource.

Such information about requests and resources can allow the scheduler to provide optimal re-allocation of requests for resources, taking into account the specifics of the request requirements and resource characteristics. Thus, a new approach to the design of new types of planners is needed in a distributed information processing system – that is. spatial planners. The article presents a theoretical rationale for a new approach to the selection and practice of a spatial planner. The article shows that application-resource relations in distributed systems can be represented in the form of a bipartite graph. And the task of distributing the request for a resource is reduced to the task of finding the maximum pairing in a weighted or unweighted bipartite graph.

Key word: *planning, dispatching, scheduling, distributed systems.*

Вступ. Ефективність роботи обчислювальної системи залежить не тільки від власної ефективності алгоритмів обробки інформації і технічних характеристик обчислювальної системи, але і від прийнятих в системі правил виконання робіт, прийому і обробки запитів користувачів.

Ефективність методів обслуговування визначається можливістю затримки або втрати заявки до обробки, а також часом знаходження заявки в системі. Залежно від типу системи управління і диспетчеризації, затримка заявок може враховуватися по загальному середньому часу затримки або по допустимому часу очікування.

Під час вивчення дисциплін обслуговування заявок передбачається, що процеси введення і обслуговування є незалежними. Заявка, яка поступає в систему, починає обслуговуватися негайно, якщо у цей момент ресурс для її обслуговування вільний. Якщо ресурс зайнятий обслуговуванням попередніх заявок, тоді залежно від типу заявки, яка поступила, і прийнятого в системі правила (дисципліни) обслуговування, заявка, що тільки що поступила, може чекати свою чергу або перервати заявку, яка виконується. У разі переривання заявки передбачається, що вона повертається в чергу, де вона чекатиме продовження перерваного обслуговування. Тривалість перебування кожної заявки у обчислювальній системі складається з часу очікування заявки і часу обслуговування машиною.

Розподіл заявок між ресурсами, які їх виконують і котрі є в наявності носить назву ***планування або диспетчеризація***. Одним із методів планування, орієнтованих на захоплення ресурсу, є метод черг. Нові заявки знаходяться у вхідній черзі, що часто зветься чергою робіт – завдань (***job queue***) та очікують звільнення ресурсу.

Аналіз алгоритмів планування та вибору заявки на виконання

Розрізняють два типи дисциплін обслуговування – *без пріоритетні* та *пріоритетні*.

У разі дисциплін без пріоритетів заявки різних типів не мають наперед встановлених пріоритетів для обслуговування і вважаються при вході в систему рівно пріоритетними. При реалізації *пріоритетних* дисциплін обслуговування окремим задачам надається привілейоване право переходу в стан виконання. Пріоритет присвоєний задачі, може бути величиною постійною, або може змінюватись в процесі її розв'язання. У деяких системах вводяться класи пріоритетів (частково впорядковані системи пріоритетів). Це як правило робиться за рахунок організації декількох черг для кожного класу. Ресурс буде представлений в першу чергу тим заявкам, котрі знаходяться в черзі (класі) з найбільш високим пріоритетом.

Витісняючи та не витісняючи алгоритми диспетчеризації

Диспетчеризація без перерозподілу процесорного часу в час виконування заявки це є ***не витісняюча багатозадачність (відносний пріоритет ВП)*** – це

такий спосіб диспетчеризації при якому активний процес виконується до тих пір поки він сам не віддасть управління диспетчеру задач для вибору із черги іншого готового до виконання процесу. При не витісняючий багатозначності механізм розподілу процесорного часу розподілений між системою та прикладними програмами. Диспетчеризація з перерозподілом процесорного часу між задачами є ***витісняючою багатозадачністю (абсолютний пріоритет АП)*** . Це такий спосіб при якому рішення про переключення процесору з виконання одного процесу на виконання іншого приймається диспетчером задач. При витісняючий багато задачності механізм диспетчеризації задач цілком зосереджений в операційній системі і програміст може писати своє програмне забезпечення не турбуючись про те, як воно буде виконуватись разом з іншими задачами. При цьому ОС виконує наступні функції: визначає момент зняття з виконання поточної задачі, зберігає її контекст; вибирає з черги готових задач наступну і запускає її на виконання наперед завантаживши її контекст.

Для однопроцесорних, а часто и для багатопроцесорних систем використовуються наступні дисципліни:

– ***Перший прийшов – перший обслуговується*** **(first IN– first OUT (FIFO)** [1].

Алгоритм обслуговування черг Firstin, Firstout (FIFO), також званий First Come First Served є найбільш простою стратегією планування процесів і полягає в тому, що ресурс передається тому процесу, котрий раніше всіх інших звернувся до нього. Коли процес потрапляє в чергу готових процесів, process control block приєднується до хвоста черги. Середній час очікування для стратегії FIFO є часто досить великим і залежить від порядку надходження процесів в чергу готових процесів. Стратегії FIFO притаманний так званий «ефект конвою». В тому випадку, коли в комп'ютері є один великий процес та декілька малих, то всі процеси збираються на початку черги готових процесів, а згодом в черзі до обладнання. Таким чином, “ефект конвою” призводить до зниження пропускної здатності як процесору, так і периферійного пристрою.

– Обслуговування в зворотному напрямку зветься LIFO – (last in – first out) і має. тіж самі недоліки як і FIFO [1].

– ***Стратегія – найбільш коротка робота (*****SHORTEST JOB FIRST – (SJF)) *[2].***

SJF – Shortest Job First. Одним із методів боротьби з «ефектом конвою» є стратегія, котра дозволяє заявці із черги виконуватися першою.

Найбільша складність у практичній реалізації SJF полягає в неможливості наперед визначити величину часу наступного обслуговування. Тому стратегія SJF часто застосовується в довгострокових (статичних) планувальниках, що обслуговують пакетний режим.

– ***Пріоритетне обслуговування черг (PQ) [3].***

Ця стратегія передбачає, що кожному процесу надається пріоритет, що визначає черговість надання йому CPU. Наприклад, стратегія FIFO передбачає, що всі процеси мають однакові пріоритети, а стратегія SJF передбачає, що пріоритет є величина, зворотна часу наступного обслуговування. Головний недолік пріоритетного планування полягає у можливості блокування на невизначено довгий строк низько пріоритетних процесів.

– ***Планування з використанням багаторівневої черги.*** **(Multilevel scheduling queue – MQS). [5]**

Ця стратегія розроблена для систем, коли заявки можуть бути легко класифіковані на декілька груп, наприклад, часто заявки розділяють на дві групи: інтерактивні (процеси переднього плану) і пакетні (фонові).

Стратегія багаторівневої черги розділяє чергу готових процесів на декілька черг, в кожній з котрих знаходяться процеси з однаковими властивостями, і кожен із котрих може плануватися індивідуальною стратегією, наприклад Round Robin стратегія для інтерактивних процесів і FIFO для пакетних процесів

– ***Планування з використанням багаторівневої черги з зворотнимі зв’язками.*** **(MFQS (multilevel feedback queue sheduling) [6].**

Звичайна багаторівнева черга не допускає переміщення заявки між чергами. Багаторівнева черга з зворотними зв'язками передбачає, що заявки при певних умовах можуть переміщатися між чергами. Розглянута стратегія є найбільш універсальною і поєднує в собі властивості всіх розглянутих раніше стратегій: FIFO,SJF, пріоритетна, Round Robin, багаторівнева черга.

– ***Алгоритм Корбато [7]***

Вважається, що тривалість виконання програми приблизно пропорційна її довжині. Принаймні, від довжини програми прямо залежить час, що витрачається на передачу програми між ОЗУ і зовнішнім ЗУ при її активізації.

Визначення номера черги, в яку поступає програма при первинному завантаженні, здійснюється по алгоритму планування Корбато: програма відразу поступає в чергу i = [log2 lp/ltk + 1], де lp – довжина програми в байтах; ltk – число байт, які можуть бути передані між ОЗУ і зовнішньою пам'яттю за час tk. Ця дисципліна дозволяє скоротити кількість системних перемикань за рахунок того, що програмам , що вимагають більшого часу рішення, надаватимуться чималі кванти часу вже при першому занятті ними ресурсу.

– *FBn системи* [8]

(Foregraund Backgraund). Алгоритм має N черг. Вхідний потік заявок поступає в першу чергу. Із черг заявки поступають на виконання. Якщо заявка за відведений квант часу не встигла завершитися, то вона повертається в чергу *i+1*, де *i* – черга з якої заявка була взята. З найбільш високим пріоритетом черга № 1. Черга *i* обслуговується, якщо порожні всі черги котрі менші за *i*. Квант часу для заявки визначається по формулі *2i-1*, заявка з останньої черги обслуговується стільки часу, скільки їй необхідно до завершення.

– ***«Карусельна» стратегія планування*** **(RR-Round Robin) [9].**

Round Robin це стратегія, коли для обслуговування визначається невеликий відрізок часу, що зветься квантом часу (10..100 мс). Черга готових заявок розглядається як кільцева. Заявки циклічно переміщаються по черзі, отримуючи CPU на деякий час, що рівний одному кванту. Новий процес додається у хвіст черги . Якщо процес не завершився в межах виділеного йому кванту часу, то його робота примусово переривається, і він переміщується в хвіст черги._Властивості Round Robin стратегії дуже залежать від величини часового кванту q. Чим більший часовий квант, тим більше Round Robin стратегія наближається до FIFO стратегії (для розглянутого прикладу, якщо q>24 мс, то -> FIFO).

– ***Алгоритм*** PSJF [9].

Алгоритм PSJF (preemptive SJN -SJN з витісненням) – поточний активний процес переривається, якщо його час, що залишився виконання більше, ніж у новоприбулого процесу. Алгоритм забезпечує ще більшу перевагу коротким процесам перед довгими. Зокрема, в ній усувається то зростання штрафних показників для найкоротших процесів, яке має місце в SJN

– ***Алгоритм*** RRSJF [9]

Модифікація алгоритму RR з переупорядочивания процесів в черзі відповідно до часом, що залишився виконання

– ***Алгоритм*** HPRN [9]

Алгоритм HPRN (highest penalty ratio next – з найбільшим штрафним показником -наступний) – алгоритм без витіснення, що забезпечує найкращі показники справедливості. Це досягається за рахунок динамічного перевизначення пріоритетів. Всякий раз при звільненні ЦП для всіх готових процесів обчислюється поточне штрафне відношення

p[i]=(w[i]+t[i])/t[i] де i-номер процесу; w[i] – час, витрачений процесом на очікування; t[i]-т час виконування процесу, наперед задана або прогнозована. Для щойно найдовшого процесу p[i]=1. ЦП віддається процесу, що має найбільше значення p[i]. Для коротких процесів HPRN забезпечує приблизно ті ж показники справедливості, що і SJN, для довгих – ближчі до FCFS. На великому діапазоні середнього часу виконування процесів показники, які забезпечуються HPRN, представляють середнє між SJN і FCFS і слабо залежать від часу виконування. Ще одна перевага HPRN в тому, що в часі очікування може враховуватися (з деякими ваговими коефіцієнтами) і очікування в інших чергах і, таким чином, забезпечується більш повне завантаження системи. Крім тогоHPRN в часі очікування може враховуватися (з деякими ваговими коефіцієнтами) і очікування в інших чергах і, таким чином, виконується більш повне завантаження систем.

– ***Алгоритм*** SRR [10]

Алгоритм SRR (selfish RR – егоїстичний RR) – метод з витісненням, що дає додаткового переваги процесам які виконуються, що дозволяє підвищити пропускну здатність. Всі процеси поділяються на дві категорії: нові і ті що вже мали час процесору (вибрані) Новими вважаються ті процеси, які не отримали ще жодного кванта часу ЦП, всі інші процеси – вибрані. 1При надходженні в систему кожному процесу дається певний пріоритет P0, однаковий для всіх процесів, який в подальшому зростає. В кінці кожного кванта часу перераховуються пріоритети всіх процесів, причому пріоритети нових процесів зростають на величину dA, а обраних – на величину dB. ЦП віддається процесу з найвищим пріоритетом, а за рівності пріоритетів – тому, який раніше поставлений в чергу.

– ***Алгоритм*** HLRR [11]

Алгоритм HLRR ("half-life round-robin"). Алгоритм напіврозпаду є модифікацією алгоритму RR. З кожним i-м процесом пов'язано деякий пріоритетне число P [i]. Чим воно менше, тим вище пріоритет процесу. Кожен новий процес отримує деякий початкове значення пріоритетного числа P0, однакове для всіх процесів. Крім того, з кожним процесом пов'язаний показчик використаного процесорного часу U [i] з вихідним значенням 0. Процес з найменшим значенням P [i] отримує квант часу Q (за однаковим значенням пріоритетних чисел ЦП віддається процесу, який ще довше).

За час кванта інтервальний таймер видає кілька сигналів-переривань з інтервалом dT. За кожним таким переривання лічильник U [i] активного (тільки активного!) Процесу збільшується на 1. Використання ЦП процесом закінчується при закінченні кванта и переході процесу в стан очікування. При цьому модифікуються лічильники процесорного часу всіх (в тому числі і неактивних) процесів: U [i] = U [i] / 2 і для всіх процесів переобчислюють пріоритетні числа: P [i] = P0 + U [i] / 2.і модифікується черга очикування процесів.

Розглянуті найбільш відомі дисципліни обслуговування заявок не враховують, що сучасні обчислювальні системи є багатомашинними розподіленими системами. Для таких систем потрібен новий підхід до систем планування та диспетчеризації. При розробці дисциплін обслуговування потрібно враховувати, що

ресурси для призначення заявок розподілені в просторі (GRID, CLOUD системи) і часто є неоднорідними.

Обґрунтування та математична постановка задачі розподілу в просторовій моделі завдань і ресурсів

У неоднорідній розподіленій системі обробки даних (НРСОД), що складається з N ресурсів, на даний момент τ є N_τ вільних ресурсів та M незалежних, готових до виконання завдань.

– Система ресурсів задана графом системи $G_R=(V_R,E_R,W_{ER})$, де

– Безліч вершин $V_R=\{R_1, R_2,..., R_N\}$, кожен елемент якого представляє один з N ресурсів системи та $R_i\in N|$ (множина натуральних чисел), i=1..N.

– Безліч дуг $E_R=\{E_1, E_2,.., E_d\}$, кожен елемент якого представляє зв'язок між двома ресурсами $Ei=\{Ri,Rj\}$, де $R_i,R_j\in V_R$ и $0\le d\le N^2$.

– Безліч ваг вершин $W_{VR}=\{WVR_1, WVR_2,..., WVR_N\}$, де $WVR_i=\{RE_i, RT_i\}$. Для $\forall i=1..N$, $RE_i\in\Re^+$ (безліч позитивних реальних чисел) є характеристика ресурсу $RT_i\in\{0$ и $\Re^+\}$ – стан ресурсу.

– Безліч ваг дуг $W_{ER}=\{WER_1, WER_2,.., WER_p\}$. Це безліч можна у вигляді деякої матриці $RC=RC[i,j]\in\Re^+$, де i=1..N і j=1..N.

– Потік M завдань, заданий безліччю $V_J=\{Job_1, Job_2,..., Job_M\}$, кожен елемент якого представляє одне з M завдань та $Job_i=\{JN_i, JE_i, JL_i, JM_i, JP_i\}$, $\forall i=1..N$:

– $JN_i\in N|$ – є номер завдання,

– $JE_i\in\Re^+$ – є обсяг роботи завдання,

– $JL_i=\{(R^1, \phi_1),.., (R^q, \phi_q)\}$, де $R^l\in V_R$ – є ресурс, з яким це завдання вимагає обміну даними, $\phi_l\in\Re^+$ – обсяг передачі, l=1..q, $q\in N|$,

– $JM_i=\{0$ или $R^i\}$ – є маска завдання, де $R^i\in V_R$ – номер ресурсу, на якому бажано виконувати дане завдання,

– $JP_i\in\Re^+$ – є пріоритетом цього завдання.

Визначення 1: Г є відображення безлічі завдань $V_J=\{Job_1, Job_2,.., Job_M\}$ на множину ресурсів $V_R=\{R_1, R_2,.., R_N\}$ графа системи $G_R=(V_R,E_R,W_{VR},W_{ER})$, якщо результат відображення $\Gamma(V_J,V_R)$ є кілька A: $A=\{a_1, a_2,.., a_n\}$, де $a_i=(R^i, J^i)$, $R^i\in V_R$, $J^i\in V_J$, i=1..n, $n\in N|$.

Позначимо $AR=\{R^1, R^2,.., R^n\}$, $AJ=\{J^1, J^2,.., J^n\}$. Отже, $|A|=|AR|\cap|AJ|$, $AR\subseteq V_R$, $AJ\subseteq V_J$.

Визначення 2: відображення Г є розподіл завдань V_J на ресурси V_R, якщо його результат $\Gamma(V_J,V_R)=A$, де $A=\{(R^1, J^1), (R^2, J^2),.., (R^n,J^n)\}$ задовольняє наступній умові: $\forall i=1..n$, $R^i\notin AR\backslash R^i$, $J^i\in AJ\backslash J^i$, где $AR=\{R^1, R^2,.., R^n\}$, $AJ=\{J^1, J^2,.., J^n\}$. Розміром цього розподілу $\Gamma(V_J,V_R)$ є число n. Тоді $\Gamma(V_J,V_R)\to A$, $n=|A|$.

Визначення 3: результат розподілу завдань на ресурсах $A=\Gamma(V_J,V_R)$ називається розкладом для даного розподілу Г. Пара $a_i=(R^i,J^i)$, i=1..n називається *призначенням ресурсу* $J^i\in V_J$, *на завдання* $R^i\in V_R$.

Визначення 4: нехай $X=\{A^1, A^2,.., A^z\}$, $z\in N|$ *– є множина результатів усіх можливих розподілів для безлічі завдань* V_J *та для безлічі ресурсів* V_R. *Тоді* $\Gamma(V_J,V_R)\equiv X$. *Розподіл завдань на ресурсах* $\Gamma(V_J,V_R)\to A^*$ *є максимальним розподілом для даних безлічі завдань* V_J *та безлічі ресурсів* V_R *якщо:*

1) $n^*=|A^*|$;

2) $n^*=\max\{|A^1|, |A^2|,.., |A^z|\}$.

Визначення 5: нехай Δ *є деяка функція від призначення* $a_s=(R^s, J^s)$ *(тобто призначення завдання* J^s *на ресурс* R^s, $R^s\in V_R$ и $J^s\in V_J$. *Тоді* $\Delta(a_s)=\Phi$ або $\Phi=\Delta(R^s, J^s)$ *та* $\Phi_i=\Delta(a_i)=\Delta(R^i, J^i)$, *де i=1..n назвемо вагою призначення* $a_i=(R^i, J^i)$ *за* Δ

Визначення 6: $суму_n$ваг усіх призначень {a1, a2,.., an} назвемо вагою D(A) розкладу A. Тобто: $D(A)=\sum_{i=1} D(a_i)$.

Визначення 7: нехай Xm={A_1, A_2,.., A_m}, m∈N|, є безліч всіх максимальних розподілів для безлічі завдань V_J та для множини ресурсів V_R. Тоді розклад A*=Γ(V_J,V_R) – оптимальний розклад розподілу (завдань V_J на ресурсах V_R) Γ за виміром Δ, якщо A* задовольняє наступним умовам:

1) A*={(R^1, J^1), (R^2, J^2),.., (R^n, J^n)} є результатом максимального розподілу даних безлічі завдань V_J і безлічі ресурсів V_R , тобто |A*|=max{|A^1|, |A^2|,.., |A^z|} (Визначення 5).

2) Вага розкладу A*={a_1, a_2,.., a_n} була максимальною з Xm={A_1, A_2,.., A_m}, тобто:

$$D[A^*]=\sum_{i=1}^{n}[a_i]=\max\{D[A_1],D[A_2],...D[A_m]\}=\max_{i=1}^{m}\{D[A_i]\}$$

Вимога: потрібно знайти оптимальне (вага за заданою функцією Δ) розклад ***A***={(R^1, J^1), (R^2, J^2),.., (R^n, J^n)}, n∈N| *максимального розподілу* Γ *(за визначенням 7) для* N_τ *вільних ресурсів (*V_R*) та M готових до виконання завдань (*V_J*).*

Визначення оптимального розподілу

– Безліч N_τ ресурсів V_R={R_1, R_2,.., $R_{N\tau}$} і M завдань V_J={J_1, J_2,..,J_M} можна представляти як безліч вершин деякого графа G . Тоді безліч неорієнтованих дуг E ={E_1, E_2,.., E_d} між вершинами графа G відповідає безлічі призначень завдання J* на ресурсі R* (приклад графа для 6 ресурсів і 6 завдань на рис. 1). Дуга E_k={R_i, J_j}, де R_i∈V_R і J_j∈V_J, k=1..d, 0≤d≤N_τ×M, між вершинами J_j та R_i існує тільки тоді, коли призначення завдання J_j на ресурсі R_i є «неможливим», тобто коли $\delta_{i,j} \le \delta_0$, де δ_0 є деяке задане число (у даному прикладі δ_0=1),

ЗАДАЧІ

РЕСУРСИ №	1	2	3	4	5	6
1	3	0	7	0	6	0
2	0	4	2	5	8	3
3	2	5	0	0	7	0
4	4	3	5	0	3	0
5	8	7	0	0	4	0
6	0	0	2	6	0	8

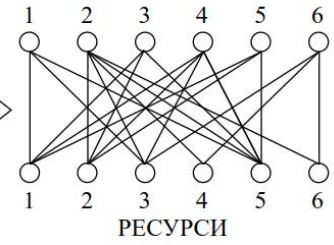

Рис. 1. Варіанти представлення відношення задачі – ресурси

Тоді, виконання другого етапу є задача призначення. Існує кілька методів для вирішення завдання призначення для зваженого дводольного графа

G=(V_R,V_J, E, WE):

де, V_R={R_1, R_2,.., $R_{N\tau}$} і V_J={J_1, J_2,.., J_M},

E ={E_1, E_2,.., E_d}, E_k={R*,J*},

де R*∈V_R и J*∈V_J , k=1..d , 0≤d≤N_τ×M.

WE={WE_1, WE_2,.., WE_d}, WE_k=Δ(E_k), де k=1..d, 0≤d≤N_τ×M.

Розв'язання задачі призначення для графа розміром N_τ×M, де N_τ≠M наводиться до розв'язання задачі призначення для графа розміром N×N, де N=max{N_τ; M}. Розв'язання задачі призначення для зваженого графа G можна призвести до вирішення задачі призначення для незваженого графа G' отриманого з графа G зняттям ваги всіх дуг.

Завдання призначення для даного випадку зводиться до вирішення задачі пошуку максимального паросполучення для зваженого або незваженого дводольного графа.

Завдання призначення у такій постановці вирішується у багатьох прикладних програмах. На вибір способу та алгоритму рішення впливає часова складність, т.к. час вирішення завдань планування, особливо при динамічному плануванні, є основним критерієм, навіть на шкоду якості одержуваного рішення. Як було зазначено, т.к. більшість завдань оперативної диспетчеризації чи динамічного планування можна звести до завдання пошуку максимального паросполучення, доцільно виконання порівняльного аналізу відомих алгоритмів з урахуванням єдиного критерію. Найбільш прийнятним критерієм є часова складність алгоритму, що дозволяє оцінити зміну часу вирішення задачі від розмірності, що є істотним для паралельних систем. Найчастіше використовувані підходи до вирішення цієї задачі це:

– пошук максимального потоку в мережі

– пошук максимального паросполучення.

Завдання про максимальний потік є одним із найбільш фундаментальних завдань у теорії потоків у мережах. Вперше це завдання було сформульовано Фалкерсоном і Данцингом 1955 року, а алгоритм її точного рішення описаний Фордом і Фалкерсоном 1956 року [12], з допомогою їх відомого алгоритму збільшуючегося шляху. З того часу багато дослідників намагалися зменшити часову складність запропонованого Фордом та Фалкерсоном алгоритму і з'явилося безліч алгоритмів для вирішення цього завдання.

Завдання формулюється наступним чином. Для заданого графа $G=(V,E)$ потрібно знайти безліч ребер $M \subseteq E$ максимальної потужності, таке, що жодні два ребра M не мають загальних кінцевих вершин. Таке формулювання завдання пошуку максимального паросполучення справедливе для незваженого графа. В іншому варіанті дані також ваги ребер і метою розв'язання задачі є знаходження паросполучення, що має найбільшу сумарну вагу. Обидві завдання викликали великий інтерес дослідників останні три десятиліття. Вони легко формулюються, апелюють до інтуїції та мають багато додатків. Ряд книг, що містять найповніший виклад питань, пов'язаних із завданням про паросполучення, написали: Форд і Фалкерсон, Берж, Пападимитріу і Стайгліц, Кофман, Липський, Харарі, Оре. Як у незваженому, і у зваженому варіантах задачі про паросполучення, ці питання значно спрощуються, якщо аналізований граф є двудольним. Відомі алгоритми пошуку максимального паросполучення засновані на теоремі Кеніга-Холла і теоремі Бержа [13]. Відповідно до теорії Бержа – «паросполучення M у графі G максимально тоді і тільки тоді, коли не існує збільшується шляху щодо M». Це призводить до того, що викунуються зайві дії в тих випадках, коли для якогось графа не існує досконалого паросполучення. Такий підхід збільшує часову складність вирішення задачі і потребує нових підходів.

Висновки.

1. Аналіз найпоширеніших дисциплін обслуговування заявок показує непридатність їх до вирішення задачі розподілу заявок за ресурсами в розподілених системах обробки інформації. Таких як глобальні GRID I CLOUD системи.

2. Виконано математичну постановку задачі розподілу заявок на ресурси, коли та заявки та ресурси просторово розподілені.

3. Показано, що в розподілених системах обробки інформації відносини заявка-ресурс можна уявити в вигляді дводольного графа, а розв'язання задачі призначення заявки на ресурс до пошуку максимального паросполучення у зваженому або не зваженому дводольному графі.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ:

1. *Z. Sazvara,* A novel mathematical model for a multi-period, multi-product optimal ordering problem considering expiry dates in a FEFO system / *Z. Sazvara, S. M. J. Mirzapour, K. Govindan, B. Bahlib* // Transportation Research Part E: Logistics and Transportation Review. Elsivier, 2016. Т. 93, Вип. September. С. 232-261.

2. Nathaniel Brooks, Shortest-Job-First (SJF): Preemptive,Non-Preemptive Example, [Електронний ресурс]. Режим доступу: https://www.guru99.com/ shortest-job-first-sjf-scheduling.html.

3. Організація черг процесів та ресурсів, Черновицкий национальный университет им. Ю. Федьковича, конспект лекций, [Електронний ресурс], Режим доступу: https://studfile.net/preview/5465773/page:29/

4. Guowang Miao, Fundamentals of Mobile Data Networks / Guowang Miao, Jens Zander, Ki Won Sung, and Ben Slimane – Cambridge University Press, ISBN 1107143217, 2016.

5. Multilevel Queue Scheduling Introduction [Електронний ресурс]. Режим доступу: https://www.geeksforgeeks.org/

6. Multilevel Feedback Queue Scheduling (MFQS). [Електронний ресурс]. Режим доступу: ttps://www.studytonight.com/operating-system/multilevel-feedback-queue-scheduling

7. Ю.Е. Лях, Алгоритм Корбато / Ю.Е. Лях, Ю.Г. Выхованец, С.М. Тетюра. Медицинская информатика: учебное пособие / 7 Донецкий национальный медицинский университет им. М. Горького [Електронний ресурс]. Режим доступу: ehttps:// studfile.net/preview/5288133/page:5/

8. Ruiguo Yu, FBN: Weakly Supervised Thyroid Nodule Segmentation Optimized by Online Foreground and Background / Ruiguo Yu, Shaoqi Yan, Jie Gao, Mankun Zhao// Ultrasound in Medicine & Biology Volume 49, Issue 9, September 2023, Pages 1940-1950.

9. Implement CPU scheduling algorithms (FCFS, SJF, RR and PSJF) [Електронний ресурс]. Режим доступу:https://github.com/Offliners/CPU-SCHEDULING-Algorithms

10. Алгоритми плануванняя процесів. [Електронний ресурс]. Режим доступу: http://repo.ssau.ru/bitstream/Metodicheskie-ukazaniya/Algoritmy-planirovaniya-processov-Elektronnyi-resurs-metod-ukazaniya-k-lab-rabote-po-kursu-Sistem-programmirovanie

11. ЄВ. Крикун, Математичне та програмне забезпечення для планування задач : магістерська дис.: 121 Інженерія програмного забезпечення, 2019.

12. Стьопкін А.В. АЛГОРИТМ ФОРДА-ФАЛКЕРСОНА / Стьопкін А.В., Пластун Д.А. State Teachers' Training University, Slovians'k, Ukraine, сб. Інформатика та методика її викладання, Випуск № 6, 2016.

13. *Douglas B. West.* Introduction to Graph Theory. 2nd. Pearson Education, Inc., 2001. С. 109–110.

REFERENSES:

1. Z. Sazvara. (2016). A novel mathematical model for a multi-period, multi-product optimal ordering problem considering expiry dates in a FEFO syste. Transportation Research Part E : Logistics and Transportation Review. Elsivier. Vol. 93, S. 232-261 [in English].

2. Nathaniel Brooks. (2020). Shortest-Job-First(SJF):Preemptive,Non-Preemptive Example, [Electronic resource]. https://www.guru99.com/ shortest-job-first-sjf-scheduling. [in English].

3. Orhanizatsiya cherh protsesiv ta resursiv, (2021). Chernovytskyy natsyonal′nyy unyversytet ym. YU. Fed′kovycha, konspekt lektsyy [Organization of queues of processes and resources], [Elektronnyy resurs], Rezhym dostupu: https://studfile.net/preview/5465773/page:29/ [in Ukrainian].

4. Guowang Miao, (2016). Fundamentals of Mobile Data Networks. Cambridge University Press, ISBN 1107143217 [in English].

5. Multilevel Queue Scheduling Introduction. (2022). https://www.geeksf

6. Multilevel Feedback Queue Scheduling (MFQS) (2020) Access mode, ttps://www.studytonight.com/operating-system/multilevel-feedback-queue-scheduling. [in English].

7. Yu.E.Lyakh. (2021). Alhorytm Korbato. [Corbato Algorithm] Medical informatics: textbook / Donetsk National Medical University named after M. Gorky. Access mode: ehttps:// studfile.net/preview/5288133/page:5/ [in Ukrainian].

8. Ruiguo Yu, FBN. (2023). Weakly Supervised Thyroid Nodule Segmentation Optimized by Online Foreground and Background. Ultrasound in Medicine & Biology Volume 49, Issue. Pages 1940-1950 [in English].

9. Implement CPU scheduling algorithms (FCFS, SJF, RR and PSJF). (2019) Access mode: https://github.com/Offliners/CPU-SCHEDULING-Algorithms [in English].

10. Process planning algorithms, (2021) [Alhorytmy planuvannyaya protsesiv].– Access mode: http://repo.ssau.ru/bitstream/ Metodicheskie-ukazaniya/ Algoritmy-planirovaniya-processov-Elektronnyi-resurs-metod-ukazaniya-k-lab-rabote -po-kursu-Sistem-programmirovanie [in Ukrainian].

11. EV Krykun. (2019) Matematychne ta prohramne zabezpechennya dlya planuvannya zadach [Mathematical and software for problem planning]: master's thesis: Software engineering [in Ukrainian].

12. Styopkin A.V. (2016). FORD-FULKERSON ALGORITHM. State Teachers’ Training University, Sloviansk, Ukraine, Sat. Informatics and its teaching methods, Issue No. 6 [in English].

13. Douglas B. West. (2001). Introduction to Graph Theory. 2nd. Pearson Education, Inc. P. 109–110 [in English].