DOI: <https://doi.org/10.15407/emodel.43.01.117>
УДК 519.876

Т.А. Узденов

Державний університет «Житомирська політехніка»
Україна, 10005, Житомир, вул. Чуднівська, 103,
тел. (063) 282 34 17; e-mail: uzdenov.taras@gmail.com

Симулятор процесу диспетчеризації задач в GRID-системах з невідчужуваними ресурсами

Проаналізовано програмні засоби, які дозволяють моделювати та симулювати процес диспетчеризації завдань в великих комп'ютерних мережах та розподілених обчислювальних системах нового програмного комплексу. Запропоновано підхід до вирішення задачі планування та метод диспетчеризації потоків задач на основі невідчужуваності обчислювальних ресурсів від їх власника. В системі можуть бути використані різні за своїми властивостями та характеристиками програмно-апаратні обчислювальні засоби, такі як кластери, суперкомп'ютери, персональні комп'ютери, ноутбуки та ін. Розроблено програмний комплекс, що дозволяє симулювати роботу GRID-системи з невідчужуваними ресурсами, а також спостерігати та досліджувати роботу різних алгоритмів в різних умовах. Описано архітектурну модель розробленого симулятора, його основні функції та можливості використання не тільки для аналізу алгоритмів диспетчеризації, але і в навчальному процесі, завдяки наявності інтуїтивно зрозумілого та інтерактивного графічного інтерфейсу, що дозволяє спостерігати за процесом розподілених обчислень на вузлах системи.

К л ю ч о в і с л о в а: алгоритм диспетчеризації, GRID-система, симулятор GRID, розподілені обчислення, планування потоків.

В наш час виникає багато задач, які для ефективного вирішення потребують великої обчислювальної потужності комп'ютерних систем, і їх виконання на одному персональному комп'ютері (ПК) є проблематичним. Є і такі задачі, для яких недостатньо обчислювальної потужності кластера. Тому об'єднують не тільки ПК, але і кластери, навіть такі, що географічно віддалені один від одного, перебуваючи в різних країнах. В результаті в середині 1990-х років з'явилося поняття GRID-системи з невідчужувальними ресурсами [1].

Одна з актуальних проблем, яка виникла перед розробниками таких систем, — вирішення задачі диспетчеризації потоків. Суть її полягає в

© Узденов Т.А., 2021

розподілі задач між вузлами системи таким чином, щоб продуктивність її була максимально великою, а час виконання — мінімальним. Проблема диспетчеризації потоків задач є досить складною і наразі не існує чіткого і однозначного її вирішення [2]. Зважаючи на це, необхідно мати програмні засоби для симулювання GRID-системи та інструментарій, який дозволяє досліджувати різні алгоритми диспетчеризації в різних умовах. Аналіз існуючих програмних засобів засвідчив, що, незважаючи на велику кількість інструментів для моделювання мереж, дуже мало є програм для моделювання процесу диспетчеризації в обчислювальному середовищі GRID-системи з невідчужуваними ресурсами [3].

Пропонується новий підхід до вирішення задачі диспетчеризації потоків в GRID-системах з невідчужуваними ресурсами і програмний комплекс, який розкриває суть цього підходу. Процес диспетчеризації, алгоритми планування та опис нового підходу і алгоритми, розроблені на його основі, наведено у роботах [4, 5].

Постановка задачі. Суть запропонованого підходу полягає в розподілі задач між ПК із збереженням принципу балансу сил. Тобто сила задачі має приблизно дорівнювати силі ПК, на якому вона буде виконуватись. Але оскільки поняття сили ПК та сили задачі є досить абстрактними, було прийнято рішення використовувати поняття потужності задачі та потужності ПК.

Будь-який ПК має такі основні характеристики: тактова частота процесора, кількість ядер, об'єм оперативної пам'яті, тип та швидкість диску, швидкість каналу зв'язку та ін. Крім того, з огляду на те, що дане дослідження проводиться для GRID-систем з невідчужуваними ресурсами, потрібно враховувати також можливість часткової загрузки ПК. Сукупність цих властивостей можна узагальнити і назвати потужністю ПК. Водночас, будь-яка задача має також різні властивості, наприклад кількість операцій, об'єм даних, розмір файлів та ін. В даному випадку всі ці властивості також узагальнюємо і вводимо поняття потужності задачі.

Метод, оснований на запропонованому підході, назвемо алгоритм диспетчеризації потоків (Flow Scheduling Algorithm (FSA)), або алгоритм балансу сил (Forces Balance Algorithm (FBA)).

А л г о р и т м:

1. Баланс сил дорівнює одиниці.
2. Існує n потоків та m процесів.
3. Вибираємо перший потік.
4. Знаходимо процес, при якому відношення першого потоку до m -го процесу максимально наближене до одиниці.
5. Відправляємо перший потік на знайдений процес.
6. Вибираємо другий потік.

7. Знаходимо процес, при якому відношення другого потоку до $(m-1)$ -го процесу максимально наближене до одиниці.
8. Відправляємо другий потік на знайдений процес.
9. Вибираємо n -й потік.
10. Знаходимо процес, при якому відношення n -го потоку до $(m-n)$ -го процесу максимально наближене до одиниці.
11. Відправляємо n -й потік на знайдений процес.

Основна складність даного підходу — пошук відношення процесу до потоку. Тому було введено поняття потужності ПК та потужності задачі. Будемо вважати, що дані характеристики відомі, тому в симуляторі ми будемо генерувати їх довільно і на їх основі розподіляти задачі по вузлам.

Отже, для підтвердження ефективності роботи запропонованого алгоритму потрібно перевірити його та порівняти з іншими алгоритмами планування.

Аналіз існуючих засобів моделювання GRID-систем. Наразі можна виділити такі програмні інструменти, які використовуються для моделювання комп'ютерних мереж та дослідження алгоритмів диспетчеризації:

- OMNET ++ — програмний пакет для моделювання роботи великих комп'ютерних мереж, здебільшого орієнтований на імітаційний комп'ютер мережі та інші розподілені системи. OMNET ++ повністю програмований і модульний. Він був розроблений для моделювання дуже великих мереж, має можливість легко відстежувати та налагоджувати імітаційну модель, має потужний графічний інтерфейс. Це робить внутрішні елементи імітаційної моделі повністю видимими для людини, яка виконує симуляцію: відображається мережева графіка і потік повідомлень, користувачеві дозволяється зазирнути в об'єкти та змінити їх в рамках моделі. Ці особливості роблять OMNET ++ хорошим об'єктом як для досліджень, так і для навчальних цілей. Модуль моделювання OMNET ++ може бути вбудованим у великі програми. OMNET ++ є відкритим джерелом, безкоштовним для некомерційного використання [6].

- Bricks — система оцінки продуктивності мережі, яка дозволяє аналізувати та порівнювати різні схеми планування на типових високопродуктивних глобальних обчислювальних установках. Призначена для моделювання глобальної мережі та різних вузлів в системі. Може імітувати різноманітну поведінку глобальних обчислювальних систем, особливо поведінку мереж та алгоритмів планування ресурсів. Bricks komponується так, що її складові можуть бути замінені для імітації різних системних алгоритмів, вона також дозволяє включити існуючі глобальні обчислювальні компоненти через зовнішній інтерфейс [7].

- MicroGrid. Основна задача даного симулятора — моделювання GRID експериментів у віртуальному середовищі. Це є віртуальне середовище, яке імітує роботу GRID-системи та підтримує Grid-програми, що використовують проміжне програмне забезпечення Globus Grid [8].

- SimGrid. Фреймворк, написаний мовою програмування C/C++, задуманий як інструмент для вивчення розподілених алгоритмів. Його сучасний інтерфейс S4U дозволяє легко оцінювати хмарні, P2P, HPC, IoT і подібні налаштування. Типове моделювання SimGrid складається з декількох елементів — «акторів», які виконують функції, що надаються користувачем. Ці елементи повинні виконувати свої обчислення, передавати дані, використовувати диск чи виконувати інші дії, чітко слідуючи за інтерфейсом S4U, що відображається в симуляторі. Ця діяльність відбувається на ресурсах (хости, посилання, диски). SimGrid передбачає час, який забирає кожна активність, і відповідно організовує «акторів», які чекають завершення цієї активності [9].

Системи Bricks та MicroGrid створено для моделювання та оцінки ефективності великих комп'ютерних мереж, а для дослідження алгоритмів диспетчеризації вони не мають достатньо функціональних можливостей і потребують значного доопрацювання. Крім того, система MicroGrid працює у віртуальному середовищі, що потребує великої апаратної потужності системи, тобто кластера чи хоча б суперкомп'ютера, а не звичайного ПК.

Найбільш потужний інструмент серед зазначених вище є саме SimGrid або його наступне покоління SimGrid 2. Даний фреймворк, розроблений саме для дослідження алгоритмів диспетчеризації в навантажених системах та мережах, постійно підтримується і розвивається. Тому він найчастіше зустрічається в наукових працях дослідників з усього світу як інструмент для дослідження нових алгоритмів та порівняння з існуючими.

Окрім позитивних властивостей SimGrid має і ряд недоліків, які стали причиною відхилення можливості використання його в даному випадку. У запропонованому новому підході до вирішення задачі диспетчеризації для GRID-систем з невідчуваними ресурсами даний фреймворк не може надати такої можливості без додаткового доопрацювання та розробки нових бібліотек. Не зовсім зрозуміло, як можна враховувати потужності окремих вузлів та потужності задач при розрахунках в алгоритмах. Також не зовсім зручна процедура запуску задачі на виконання при симуляції постійної зміни кількості вузлів та їхньої потужності. Для цього потрібно кожен раз конфігурувати систему та компілювати програму, що в даному випадку не дуже зручно. Крім того, є потреба в експери-

ментах відкинути все лишнє в системі та зосередитись саме на алгоритмі в чистому вигляді.

Незважаючи на існуюче твердження «навіщо вигадувати велосипед?», відомо, що інколи на вивчення та налаштування чужого програмного коду можна витратити набагато більше часу, ніж на написання своєї бібліотеки чи програми. Тому було прийнято рішення створити програму, яка б могла симулювати роботу GRID-системи з невідчужуваними ресурсами, використовуючи різні алгоритми, а головне, щоб можна було показати особливість нового підходу.

Розробка програмного комплексу для симуляції роботи GRID-системи з невідчужуваними ресурсами та дослідження алгоритмів диспетчеризації. Для створення програми було використано Windows Communication Foundation (WCF) та мова програмування C#. WCF — це платформа для створення сервісно-орієнтованих додатків, за допомогою якої можна надіслати дані у спільні повідомлення від однієї кінцевої точки служби до іншої. Кінцева точка служб може бути частиною постійно доступної служби, розміщеної в IIS, або службою, розміщеною в додатках. Кінцева точка може бути клієнтом служб, яка запитує дані з кінцевих пунктів служби. Повідомлення можуть бути як прості (один символ або слово), що передаються як XML (eXtensible Markup Language), так і складні, як потік двійкових даних [10].

В документації WCF засвідчено, що на базі даної платформи можна будувати різні сервіси для різних задач, наприклад:

- обробка бізнес-транзакцій;
- мережевий чат;
- надання актуальних даних інших сервісів;
- моніторинг даних або станів інших служб та подання їх у потрібному вигляді;
- зображення робочого процесу, реалізованого за допомогою Windows Workflow Foundation як служби WCF, та ін.

В результаті аналізу можливостей WCF зроблено висновок, що дана платформа цілком підходить для вирішення поставленої задачі, і створено систему, яка складається з трьох частин — Client, Server, Host.

1. Client — програма, створена на основі платформи WPF [11], яка симулює роботу ПК, або вузла в GRID-системі: виконує функції підключення (відключення) до Host, приймає задачу від Server, виконує її, відправляє результат. Додатково реалізована можливість спостерігати за її роботою, тобто користувач може бачити, коли Client працює. В системі може бути задіяно безліч клієнтів, на одному ПК також може бути декілька клієнтів. Кількість задіяних клієнтів на одному ПК обмежується

Рис. 1. Схема компонентів симулятора GRID-системи та їх основні функції

лише характеристиками ПК, здебільшого об'ємом оперативної пам'яті. Наприклад, на ПК з характеристиками intel Core i7-7500U (2.7—3.5 ГГц) / RAM 24 ГБ / SSD 512 МБ / NVidia GeForce 940MX, 2 ГБ вдавалось запустити одночасно більше 100 клієнтів.

2. Server — програма, як і Client, створена на основі платформи WPF, що симулює роботу сервера. Це є головна програма в системі. Вона виконує наступні функції:

запускає Host, до якого буде здійснюватись підключення (відключення) клієнтів та через який буде передаватись інформація;

здійснює підключення (відключення) до Host;

генерує задачі на виконання, при цьому можна задавати кількість задач, а також мінімальний та максимальний об'єм задачі;

відтворює список задач, згенерованих автоматично, а також їх статус на кожному етапі обчислень та час виконання алгоритмом;

веде обрахунок часу виконання кожної задачі та обраховує загальний час роботи по кожному алгоритму;

відтворює підключених до Host клієнтів, яких можна використовувати для обчислень, а також статус клієнта (зайнятий або вільний);

для зручності роботи реалізовано можливість запуску Client, попередньо задавши кількість вузлів, яку потрібно запустити;

довільно змінює потужність вузлів, реалізуючи тим самим властивість невідчужуваності та різнорідності GRID-системи, при цьому можна задавати мінімальну та максимальну потужність вузлів;

реалізує механізм вибору алгоритму, за яким буде проводитись обчислення;

реалізує процес тестування, при якому всі згенеровані задачі обчислюються на вузлах кожним з алгоритмів, що пропонуються системою, після чого результати можна зберегти в Excel файлі для подальшого аналізу та досліджень;

в одній системі повинен бути тільки один Server.

Рис. 2. Візуальний інтерфейс компонента Host (а) і компонента Server (б)

3. Host — проста консольна програма [12], реалізує WCF контракт на обслуговування [13] для Client та Server та запускає Host, який реалізує механізм зв'язку між вузлами системи та сервером, слідкує за підключенням (відключенням) вузлів до системи та інформує сервер про зміну стану того чи іншого вузла. Також Host реалізує механізм передачі даних між вузлами та сервером.

На рис. 1 схематично зображено всі основні компоненти системи та основні функції, які вони виконують. Даний симулятор має візуальний інтерфейс та різні налаштування для дослідження (рис. 2, 3). На рис. 3 схематично зображено три клієнти, незважаючи на те, що в даному випадку їх підключено 20.

Для проведення тестування необхідно виконати наступні дії:

1. Запустити програму Server.exe з правами адміністратора (права адміністратора потрібні, щоб запустився Host).

2. У відкритому вікні (рис. 2, б) натиснути кнопку Run Host, в результаті чого буде здійснено запуск служби WCF та Host.exe.

Рис. 3. Візуальний інтерфейс компонентів Client

Рис. 4. Інформативні області програми Server

3. Здійснити підключення до Host, натиснувши кнопку Connect.

4. В області 1 (рис. 2, б) необхідно задати кількість задач, які будуть згенеровані програмою, а також їх мінімальний та максимальний об'єм (задається в секундах). Також є можливість очистити задачі.

5. В області 2 (рис. 2, б) потрібно задати кількість Client, які потрібно запустити, після цього можна довільно змінити потужності запущених Client, перед цим задавши максимальне та мінімальне значення в області 3 (рис. 2, б). Потужність 1 відповідає найбільш потужному ПК в GRID-системі, а далі вона зменшується на 10% на кожному кроці — до 0.1. На даному етапі цього достатньо, щоб побачити зміну поведінки алгоритмів при зміні потужності вузлів.

6. Після проведення всіх налаштувань необхідно запустити тест, натиснувши кнопку Start testing.

Система почне автоматично виконувати завдання, використовуючи для їх розподілу по ПК різні алгоритми по черзі. Спочатку виконуються всі завдання першого алгоритму диспетчеризації. При цьому обліковується час виконання кожної задачі, після чого підраховується загальний час роботи, від початку запуску до виконання всіх завдань. Потім ті ж самі задачі подаються на виконання за іншим алгоритмом, за яким вони розподіляються на вузли. При цьому також виконується облік часу. Така процедура продовжується до виконання тестом останнього алгоритму. В програмі реалізована можливість запуску роботи для виконання лише якогось конкретного алгоритму. Для цього необхідно вибрати алгоритм, який цікавить, і натиснути кнопку Send.

На рис. 4 показано шість областей у вікні програми Server, в які під час тестування виводиться інформація про результати роботи симулятора. У областях 1—6 відображено інформацію про загальний час (в секундах), витрачений на виконання всіх задач, що згенерувала система, або об'єм всієї роботи, яку необхідно виконати:

- 1 — сумарний об'єм всіх задач, що необхідно вирішити;
- 2 — сумарний час виконання всіх задач для кожного алгоритму;
- 3 — область, в яку виводяться повідомлення про початок та кінець обчислень, а також інформація, який алгоритм працює на даний момент;
- 4 — список всіх задач, їх об'єм та потужність, статус (виконана чи ні), час виконання за кожним алгоритмом;
- 5 — після завершення виконання тесту в дану область з області 2 надходить зведена інформація про час роботи кожного алгоритму, після чого в області 2 інформація оновлюється для проведення нового тесту;
- 6 — інформація про підключені до системи вузли, їх потужність та статус (вільні чи зайняті роботою).

Після запуску тестування всі кнопки та елементи вибору налаштувань стають неактивними до моменту завершення тестування. Результати тесту можна зберегти в файлі Exel, натиснувши кнопку Export to Exel. В даному файлі зберігається вся інформація про виконаний тест, а саме кількість вузлів та їх потужність, кількість задач та їх потужність, сумарний час виконання всіх задач та час виконання кожної задачі за кожним алгоритмом, задані настройки для мінімальної та максимальної потужностей ПК, заданий мінімальний та максимальний об'єм задачі, а також сумарні результати обчислень всіх задач за кожним алгоритмом диспетчеризації.

На рис. 5 наведено таблицю результатів тестування. Структура таблиці дозволяє далі працювати з нею. При проведенні нового тесту інформація додається до вже існуючої, а тесту присвоюється ідентифікатор, що не потребує створення нового файлу.

Рис. 5. Загальний вигляд результатів тестування в файлі MS Excel

Рис. 6. Вигляд поданих до виконання програм на двох екранах

Одною з позитивних якостей даного симулятора є можливість спостерігати, як працює система, тобто добре видно, як вузли приймають задачі та виконують їх. Цю функцію реалізовано за допомогою таймера зворотного відліку часу. Ця особливість робить зручним даний комплекс програм не тільки для досліджень, але і в навчальному процесі, для кращого розуміння розподілених обчислень та GRID-систем в цілому.

На рис. 6 показано вигляд запущеного симулятора на одному ПК з підключеним до нього монітором, тобто на двох екранах з розширенням 1920×1080. В даному випадку без особливих труднощів розміщено 15 Client та один Server.

Висновки

Аналіз існуючих програмних засобів моделювання розподілених систем та комп'ютерних мереж, що симулюють роботу GRID-систем з невідчуваними ресурсами, показав, що наразі є досить мало засобів, які надають можливість досліджувати алгоритми диспетчеризації, а відомі, наприклад фреймворк SimGrid, не можуть без додаткової розробки функціоналу надати можливість дослідити ефективність методу диспетчеризації, що пропонується. Розроблений програмний комплекс дозволяє показати роботу запропонованого методу та порівняти результати його роботи з іншими, загальновідомими, методами диспетчеризації. За допомогою даного програмного комплексу можна проводити експерименти та досліджувати роботу алгоритмів, змінюючи кількість та розмір задач, кількість та потужність ПК. Реалізований візуальний інтерфейс наглядно показує, як працює GRID-система. Даний комплекс програм можна використовувати не тільки для наукових досліджень, але і для навчального процесу.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Singh M. An overview of grid computing. Paper presented at the Proceedings International Conference on Computing, Communication, and Intelligent Systems (ICCCIS 2019). January, 2019, pp. 194—198. Doi:10.1109/ICCCIS48478.2019.8974490.
2. Кропивницька В.Б., Клим Б.В., Романчук А.Г., Слабінога М.О. Дослідження алгоритмів диспетчеризації в комп'ютерних системах // Розвідка та розробка нафтових і газових родовищ, 2011, № 2(39), с. 93—105.
3. Buyya R., Murshed M. Gridsim: a toolkit for the modeling and simulation of distributed resource management and scheduling for grid computing // Concurrency and computation: practice and experience, 2002, Vol. 14, pp. 1175—1220.
4. Данильченко О.М., Узденов Т.А. Процес диспетчеризації для GRID-системи з невідчуваними ресурсами // Вісник ЖДТУ. Серія «Технічні науки», 2012, №2(61), с. 147—154.
5. Uzdenov, T. New Task Scheduling Algorithm for GRID Systems with Non — Alienable Resources. // Systems, Decision and Control in Energy II. Studies in Systems, Decision and Control, 202, vol. 346. Doi: 10.1007/978-3-030-69189-9_12.

6. Varga A. The OMNeT++ discrete event simulation system. Proceedings of the European Simulation Multiconference (ESM). Prague, Czech Republic, 6-9 June 2001. Режим доступа <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.331.1728&rep=rep1&type=pdf>
7. Overview of a Performance Evaluation System for Global Computing Scheduling Algorithms. Режим доступа <http://ninf.apgrid.org/takefusa/bricks/contents/hpdc99.html>.
8. Song H., Liu X., Jakobsen D., et al. The MicroGrid: A scientific tool for modeling computational Grids. Proceedings of IEEE Supercomputing (SC 2000). Dallas, TX, 4–10 November 2000. Режим доступа <https://ieeexplore.ieee.org/document/1592766>
9. Simulating Algorithms — SimGrid documentation. Режим доступа https://simgrid.org/doc/latest/Tutorial_Algorithms.html
10. What Is Windows Communication Foundation — WCF | Microsoft Docs. Режим доступа <https://docs.microsoft.com/en-us/dotnet/framework/wcf/whats-wcf>
11. What is WPF? — Visual Studio | Microsoft Docs. Режим доступа <https://docs.microsoft.com/en-us/visualstudio/designers/getting-started-with-wpf?view=vs-2019>
12. Tutorial: Create a simple C# console app in Visual Studio. Режим доступа <https://docs.microsoft.com/en-us/visualstudio/get-started/csharp/tutorial-console?view=vs-2019>
13. Windows Communication Foundation service contract — WCF. Режим доступа <https://docs.microsoft.com/en-us/dotnet/framework/wcf/how-to-define-a-wcf-service-contract>

Отримано 18.12.2020

REFERENCES

1. Singh, M. (2019), “An overview of grid computing”, *Proceedings of the 2019 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*, January, 2019, pp. 194-198, DOI:10.1109/ICCCIS48478.2019.8974490.
2. Kropyvnytska, V.B., Klim, B.V., Romanchuk, A.G. and Slabinoga, M.O. (2011), “Investigation of scheduling algorithms in computer systems”, *Rozvidka ta rozrobka naftovykh i hazovykh rodovyshch*, Vol. 2, № 39, pp. 93-105.
3. Buyya, R., Murshed, M. (2002), “Gridsim: a toolkit for the modeling and simulation of distributed resource management and scheduling for grid computing”, *Concurrency and computation: practice and experience*, Vol. 14, pp. 1175-1220.
4. Danilchenko, O.M. and Uzdenov, T.A. (2012), “The process of task scheduling in geographically distributed computer systems with non-alienable resources”, *Visnyk ZHDTU, ser. «Tekhnichni nauky»*, Vol. 2, № 61, pp.147-154.
5. Uzdenov, T. (2021), *Systems, Decision and Control in Energy II. Studies in Systems, Decision and Control* (in press).
6. Varga, A. (2001), “The OMNeT++ discrete event simulation system”, *Proceedings of the European Simulation Multiconference (ESM)*, Prague, Czech Republic, June 6-9, 2001, available at: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.331.1728&rep=rep1&type=pdf>.
7. Overview of a Performance Evaluation System for Global Computing Scheduling Algorithms, available at: <http://ninf.apgrid.org/takefusa/bricks/contents/hpdc99.html>.
8. Song, H., Liu, X., Jakobsen, D., Bhagwan, R., Zhang, X., Taura, K. and Chien, A. (2000), “The MicroGrid: A scientific tool for modeling computational Grids”, *Proceedings of IEEE Supercomputing*, USA, Dallas, TX, November 4-10, 2000, available at: <https://ieeexplore.ieee.org/document/1592766>.
9. Simulating Algorithms - SimGrid documentation, available at: https://simgrid.org/doc/latest/Tutorial_Algorithms.html.

10. What Is Windows Communication Foundation – WCF?, available at: <https://docs.microsoft.com/en-us/dotnet/framework/wcf/whats-wcf>.
11. Windows Communication Foundation service contract – WCF, available at: <https://docs.microsoft.com/en-us/dotnet/framework/wcf/how-to-define-a-wcf-service-contract>.
12. Tutorial: Create a simple C# console app, available at: <https://docs.microsoft.com/en-us/visualstudio/get-started/csharp/tutorial-console?view=vs-2019>.
13. Windows Communication Foundation service contract – WCF, available at: <https://docs.microsoft.com/en-us/dotnet/framework/wcf/how-to-define-a-wcf-service-contract>.

Received 18.12.2020

T.A. Uzdenov

SIMULATOR OF TASK SCHEDULING IN GEOGRAPHICALLY DISTRIBUTED COMPUTER SYSTEMS WITH NON — ALIENABLE RESOURCES

The software tools that allow to model and simulate the process of task scheduling in large computer networks and distributed computer systems are analyzed. It is reasonably stated why there is a need to develop a new software package. A brief description of the new approach proposed by the author to solve the problem of planning and method of task flow scheduling, given the inalienability of computing resources from their owner and the fact that the system can use different properties and characteristics of software and hardware, whether clusters, supercomputers, personal computers, laptops, etc. A software package has been developed that allows simulating the operation of a geographically distributed computer system with inalienable resources, as well as observing and investigating the behavior of different algorithms under different conditions. Describes the architectural model of the developed simulator, its main functions and possibilities of use not only for the analysis of scheduling algorithms, but also in educational process, thanks to presence of the intuitively clear and interactive graphic interface allowing to observe process of distributed calculations on system nodes.

Keywords: scheduling algorithms, GRID-system, GRID simulator, distributed computing, flows scheduling.

УЗДЕНОВ Тарас Амурович, аспірант, Державний університет «Житомирська політехніка». У 2007 р. закінчив Житомирський державний технологічний університет. Область наукових досліджень — розподілені обчислення, планування завдань, GRID-системи.