НАЦІОНАЛЬНА
АКАДЕМІЯ НАУК УКРАЇНИ
ІНСТИТУТ ПРОГРАМНИХ СИСТЕМ

ПРОБЛЕМИ ПРОГРАМУВАННЯ

науковий журнал

№ 4 ***жовтень - грудень*** ***2023***

Заснований у березні 1999 р.

ЗМІСТ

Свідоцтво про державну реєстрацію КВ № 7490 від 01.07.2003

Науковий журнал “Проблеми програмування” занесений до переліку наукових фахових видань України, в яких можуть публікуватися основні результати дисертаційних робіт.

ISSN 1727-4907

NATIONAL ACADEMY
OF SCIENCES OF UKRAINE
INSTITUTE OF SOFTWARE SYSTEMS

PROBLEMS OF PROGRAMMING

scientific journal

№ 4 ***October – December*** ***2023***

Founded in March, 1999

CONTENT

УДК 004.4 http://doi.org/10.15407/pp2023.04.003

В.І. Шинкаренко, О.*В. Макаров*

ДОСЛІДЖЕННЯ ЕФЕКТИВНОСТІ ДЕЯКИХ ДЕТЕРМІНОВАНИХ МЕТОДІВ ПЕРЕДОБРОБКИ СОРТУВАННЯ ДАНИХ

Для перевірки гіпотези щодо зменшення часу сортування алгоритмами різної обчислювальної складності здійснено експерименти. Опрацьовано декілька ідей із детермінованої передобробки масивів даних для алгоритмів сортування. Запропоновані відповідні алгоритми: швидка передобробка – передбачення індексу елемента у відсортованому масиві і перестановка, передобробка з пам'яттю – передбачення і перестановка із запам'ятовуванням раніше встановлених елементів, передобробка із розворотом – розвертання послідовностей елементів, відсортованих у зворотному порядку. Також запропоновані блочні варіації швидкої та передобробки з пам'яттю, які виконуються для частин масиву заданої довжини. Встановлено, що більшу ефективність передобробки вони показують разом із алгоритмами сортування, які значно прискорюються на відсортованих (або майже відсортованих) масивах даних. Блочні передобробки можуть виконуватися швидше через можливість уникнення промахів кешу (cache miss), однак показують нижчий відсоток впорядкованості масиву. Виконані експерименти з оцінки ефективності різних алгоритмів сортування після і разом із запропонованими передобробками.

Ключові слова: алгоритм сортування, сортування, часова ефективність, комбінований алгоритм, передобробка.

Вступ

Ера великих даних (Big Data) відкриває нові можливості, але має й більше проблем для алгоритмів сортування. Класичні алгоритми сортування не можуть пристосуватися до вибухового зростання даних. Існує велика кількість алгоритмів сортування за різноманіттям сценаріїв застосування, пристроїв для зберігання даних і стратегій удосконалення.

Алгоритми сортування класифікуються за різними показниками часової ефективності, стабільності і вимог до виділення додаткової пам'яті. Прийнято оцінювати алгоритми за показником асимптотичної обчислювальної складності O [1]. Найбільш популярні алгоритми сортування, такі як швидке сортування (Quick sort) чи сортування злиттям (Merge sort), мають обчислювальну складність у середньому $O(n * log(n))$. Однак на невеликих наборах даних сортування вставками (Insertion sort) з обчислювальною складністю $O(n^2)$ працює швидше.

Ефективні реалізації зазвичай використовують гібридний алгоритм, що поєднує асимптотично ефективний алгоритм для загального сортування із сортуванням вставками для невеликих масивів у нижній частині рекурсії. Використовуються і складніші варіанти, такі як, Timsort [2] – попередній аналіз даних та пошук відсортованих послідовностей, розбиття масиву на відсортовані ділянки за допомогою сортування вставками і модифікований алгоритм злиття в один відсортований масив. Або Introsort [3], що починає із швидкого сортування, потім переходить на Heapsort, коли глибина рекурсії перевищує заданий рівень, і закінчує сортуванням невеликих послідовностей за допомогою сортування вставками.

Досліджувалися можливі модифікації існуючих алгоритмів для покращення їхньої часової ефективності. Наприклад, у [4] використовувалось сортування вставками із розбиттям масиву на дві відсортовані частини і одну невідсортовану, з якої вибиралися елементи. У [5] досліджувалось швидке сортування з різною кількістю pivot елементів.

Попередню обробку можна представити як аналіз і деяку перестановку елементів масиву. Залежно від основного алгоритму сортування попередня обробка може робити перестановки, які зменшують час сортування. Або запобігають виник-

ненню найгірших ситуацій, що призводять до погіршання часової ефективності алгоритму. В даній праці запропоновано декілька алгоритмів попередньої обробки масивів сортованих даних.

У [6] досліджувалися методи стохастичної передобробки. Випадкові елементи, вибрані з усього масиву або із його половин, певну кількість разів (кратну розміру масиву) порівнюються і, якщо вони йдуть у зворотньому порядку, міняються місцями.

У [7] розглядається метод стохастичної передобробки, в якому випадкові елементи міняються місцями спочатку в межах половин вхідного масиву, а потім ще раз у межах цілого масиву. Також представлений детермінований метод передобробки, в якому порівнюються елементи з кінця та початку масиву і, якщо вони стоять у зворотному порядку, виконується перестановка, а індекси порівнюваних елементів зміщуються до середини масиву. Процес повторюється для усіх елементів масиву, а потім рекурсивно для підмасивів.

Показники ефективності передобробки

У цьому дослідженні ефективність передобробки оцінювалась за декількома показниками.

Поліпшення впорядкованості масиву визначається за показником невпорядкованості масиву [6] після закінчення одинарних або серійних перестановок:

$$U = \frac{1}{N}\left(\sum_{i=0}^{N}\frac{|j-i|}{max(\,i, N-i)}\right) * 100\%, \quad (1)$$

де j – індекс елемента у відсортованому масиві, i – індекс того ж елемента у невпорядкованому масиві, N – загальна кількість елементів масиву. Тут і далі індексація масивів починається з нуля. Якщо у масиві є декілька однакових елементів під індексами i_1 та i_2, для них будуть використані відповідні індекси у відсортованому масиві j_1 та j_2. Зазначимо, що у разі $i_1 < i_2$, тоді $j_1 < j_2$, як за умови стабільного сортування.

Для визначення найбільш ефективного алгоритму попередньої обробки масиву використовується формула:

$$Ef = \left(\frac{U_0 - U_p}{U_0}\right) * 100\% \quad (2)$$

де U_0 – показник невпорядкованості вхідного масиву, U_p – показник невпорядкованості масиву після виконання передобробки.

Часова ефективність алгоритмів оцінюється за S показником [8]. Якщо V^* – множина можливих значень обсягу даних, T^* – множина можливих типів даних, X^* – множина можливих значень даних, Ψ^* – множина можливих програмних середовищ, $\Re^*$ – множина архітектур ЕОМ, на яких можлива реалізація алгоритмів, тоді для порівняння часової ефективності альтернативних обчислювальних алгоритмів визначається ступінь переваги одного (і-го) алгоритму над іншим (j-им) на обмежених множинах, що є підмножинами зазначених вище множин $\bar{V} \subseteq V^*$, $\bar{T} \subseteq T^*$, $\bar{X} \subseteq X^*$, $\bar{\Psi} \subseteq \Psi^*$, $\bar{\Re} \subseteq \Re^*$:

$$SUP_{ij}\left|\bar{V}, \bar{T}, \bar{X}, \bar{\Psi}, \bar{\Re} = \right. \quad (3)$$

$$= \frac{1}{N}\sum_{\forall v \in \bar{V}}\sum_{\forall t \in \bar{T}}\sum_{\forall x \in \bar{X}}\sum_{\forall \psi \in \bar{\Psi}}\sum_{\forall r \in \bar{\Re}} g_{ij}(v,t,x,\psi,r),$$

де N – загальна кількість виконань, g_{ij} – ступінь переваги і-го алгоритму над j-им в точці, визначається як:

$$g_{ij}(v,t,x,\psi,r) = \frac{\tau_j(v,t,x,\psi,r) - \tau_i(v,t,x,\psi,r)}{\max\left(\tau_i(v,t,x,\psi,r), \tau_j(v,t,x,\psi,r)\right)}, \quad (4)$$

де $\tau_j(v,t,x,\psi,r)$ – час виконання j-го алгоритму, $\tau_i(v,t,x,\psi,r)$ – час виконання і-го алгоритму.

В експериментах змінюються розміри масивів даних ($\bar{V} \subseteq V^*$) і значення елементів ($\bar{X} \subseteq X^*$). Тип даних, програмна середа та архітектура ЕОМ залишаються постійними і можуть бути видалені з рівняння. Кінцева формула для визначення ступеня переваги алгоритму має вигляд:

$$g_{ij}(v,t) = \frac{\tau_j(v,x) - \tau_i(v,x)}{\max\left(\tau_i(v,x), \tau_j(v,x)\right)}, \quad (5)$$

Експериментально статистична оцінка S показника, як обґрунтовано у [8], обчислюється так:

$$S_{ij} = \frac{1}{N}\sum_{k=1}^{N} g_{ij}(v, x_k) \qquad (6)$$

де N – кількість експериментальних випробувань алгоритму, v – фіксований у конкретному експерименті розмір масиву, x_k – випадковим чином сформований масив у k-тому випробуванні. В експериментах використовується N рівне 51.

Детерміновані методи передобробки

Розглянемо запропоновані авторами методи передобробок, що використовуються для покращення часової ефективності алгоритмів сортування.

Швидка передобробка (QP). Основна ідея швидкої передобробки полягає у спробі «передбачити» позицію елемента в масиві. Знаходиться мінімальне та максимальне значення елементів масиву. Для кожного елемента розраховується передбачуване місце (індекс) у відсортованому масиві

$$i = \frac{(x_i - \min\limits_i x_i)}{(\max\limits_i x_i - \min\limits_i x_i)} * (N-1), \qquad (7)$$

де x_i– поточний елемент масиву, N – кількість елементів у масиві. Виконується перестановка поточного елемента з елементом за передбаченим індексом. Операція повторюється аж доки для поточного елемента не буде передбачений його ж індекс, тобто, коли елемент уже стоїть на передбаченому місці, або елемент за передбаченим індексом дорівнює поточному. Кількість передбачень для поточного місця елемента не може перевищувати наперед заданого M (це дозволяє уникнути зациклювання).

Після передобробки деяка кількість елементів опиниться на місцях, відповідних їхнім місцям у відсортованому масиві. Таким чином зменшується кількість перестановок елементів, що буде виконана безпосередньо алгоритмом сортування.

Покажемо на масиві випадкових цілих чисел у діапазоні від 0 до 9 роботу передобробки. Визначаємо мінімальний і максимальний елементи масиву. Вони дорівнюють відповідно 0 і 9. Максимальний індекс дорівнює N-1 = 9. Починаємо передобробку з першого елемента масиву. Жирним шрифтом виділено поточний елемент. Затіненням жовтого кольору показано передбачене місце для поточного елемента. Праворуч від масиву на кожному кроці наведені значення показника невпорядкованості масиву (U_i).

6	0	4	4	1	3	8	9	2	5	U_0=39.5
8	0	4	4	1	3	6	9	2	5	U_1=38.8
2	0	4	4	1	3	6	9	8	5	U_2=25.3
4	0	2	4	1	3	6	9	8	5	U_3=24.8
1	0	2	4	4	3	6	9	8	5	U_4=17.1
0	1	2	4	4	3	6	9	8	5	U_5=14.9
0	**1**	2	4	4	3	6	9	8	5	U_6=14.9
0	1	**2**	4	4	3	6	9	8	5	U_7=14.9
0	1	2	**4**	4	3	6	9	8	5	U_8=14.9
0	1	2	4	**4**	3	6	9	8	5	U_9=14.9
0	1	2	4	4	**3**	6	9	8	5	U_{10}=14.9
0	1	2	3	4	**4**	6	9	8	5	U_{11}=7.9
0	1	2	3	4	4	**6**	9	8	5	U_{12}=7.9
0	1	2	3	4	4	6	**9**	8	5	U_{13}=7.9
0	1	2	3	4	4	6	**5**	8	9	U_{14}=3.1
0	1	2	3	4	5	6	**4**	8	9	U_{15}=6.5
0	1	2	3	4	5	6	4	**8**	9	U_{16}=6.5
0	1	2	3	4	5	6	4	8	**9**	U_{17}=6.5

Рис.1. Приклад швидкої передобробки (QP)

Передобробка з пам'яттю (PM). Відмінність від швидкої передобробки полягає у запам'ятовуванні передбачених індексів, у яких була виконана перестановка. Алгоритм не намагається передбачати ін-

декс для того ж самого елемента декілька разів і для поточного елемента буде знайдено наступний ще непередбачений індекс.

У передобробці з пам'яттю додатково використовується окремий масив булевих змінних. Після того, як у передбачений індекс p встановлюється елемент масиву, булевий флаг за індексом p встановлюється в значення true. Булеві значення, що відповідають кожному елементу масиву, відобразимо під кожним елементом. Напочатку усі булеві флаги дорівнюють false, що відображається знаком "-". Після встановлення елемента булеве значення зміниться на true ("+"). Жирним виділимо поточний елемент, затіненням жовтого кольору – елемент під передбаченим індексом, зеленого – елемент переставлений за передбаченим на попередньому кроці індексом. Праворуч від масиву на кожному кроці записані значення невпорядкованості масиву (U_i).

6	0	4	4	1	3	8	9	2	5
-	-	-	-	-	-	-	-	-	-

U_0=39.49

8	0	4	4	1	3	6	9	2	5
-	-	-	-	-	-	+	-	-	-

U_1=38.82

2	0	4	4	1	3	6	9	8	5
-	-	-	-	-	-	+	-	+	-

U_2=25.32

4	0	2	4	1	3	6	9	8	5
-	-	+	-	-	-	+	-	+	-

U_3=24.82

1	0	2	4	4	3	6	9	8	5
-	-	+	-	+	-	+	-	+	-

U_4=17.06

0	1	2	4	4	3	6	9	8	5
-	+	+	-	+	-	+	-	+	-

U_5=14.95

0	1	2	**4**	4	3	6	9	8	5
+	+	+	-	+	-	+	-	+	-

U_6=14.95

0	1	2	**3**	4	4	6	9	8	5
+	+	+	-	+	+	+	-	+	-

U_7=7.85

0	1	2	3	4	4	6	**9**	8	5
+	+	+	+	+	+	+	-	+	-

U_8=7.85

0	1	2	3	4	4	6	**5**	8	9
+	+	+	+	+	+	+	-	+	+

U_9=3.09

0	1	2	3	4	4	6	5	8	9
+	+	+	+	+	+	+	+	+	+

U_{10}=3.1

Рис. 2. Приклад передобробки з пам'яттю

Потребує пояснення вибір місця і перестановка на сьомому кроці, тому що для елемента «4» передбачене місце під індексом 4, де стоїть елемент «4», але у масиві флагів стоїть «+». Тому вибирається наступне після передбаченого місце, в якому флаг дорівнює «-». Таке місце знаходиться під індексом «5», за яким стоїть елемент «3».

Блочна локальна передобробка. Ідея блочної локальної передобробки полягає в розбитті масиву на блоки певної довжини і застосуванні алгоритмів, описаних раніше на кожному підмасиві. Маємо дві комбінації алгоритмів: блочна локальна швидка передобробка (BLQP) і блочна локальна передобробка з пам'яттю (BLPM). У результаті початковий масив буде мати N частково (або повністю) відсортованих підмасивів. Масив після локальної передобробки матиме «форму пилки». За умови правильного вибору довжини блоку, меншої за довжину кешу процесора, поділеної на розмір елементів у сортованому масиві, можна значно зменшити кількість (або взагалі уникнути) промахів кешу (cache miss). У разі, якщо розмір блока буде дуже великим, отримаємо багато промахів кешу і зменшення часової ефективності роботи алгоритму.

Блочна глобальна передобробка. Глобальна передобробка також включає розбиття масиву на блоки певної довжини і застосування швидкої (BGQP) та передобробки з пам'яттю (BGPM) до підмасивів. Відмінність від локального алгоритму полягає у тому, що для кожного елемента масиву передбачається глобальний індекс, тобто приблизне місце, на якому стояв би даний елемент у відсортованому масиві. Після передбачення елемент може бути переставлений, тільки якщо передбачений індекс не виходить за межі поточного блока. Такий підхід також дозволяє уник-

нути промахів кешу (cache miss). Але приблизна вірогідність того, що елемент буде переставлений, дорівнює $1/K$ (де K – кількість блоків), для масиву рівномірно розподілених випадкових чисел.

Попередня обробка із розворотом (SR). В останньому варіанті попередньої обробки виконується прохід масивом, й усі послідовності, відсортовані у зворотному напрямку, розвертаються. Тож максимальна довжина послідовностей, відсортованих у зворотному напрямку, буде дорівнювати 2 (кінець попередньої і початок наступної відсортованої послідовностей). Найбільший вплив даний вид попередньої обробки буде мати для швидкого сортування (Quick sort), для якого вхідний масив, відсортований у зворотному порядку, є спеціальним випадком, де обчислювальна складність зростає до $O(n^2)$.

Наведемо приклад роботи попередньої обробки із розворотом на масиві випадкових цілих чисел від 0 до 9. Послідовності, відсортовані у зворотному порядку, перед розворотом виділені затіненням жовтого кольору, після – зеленого кольору.

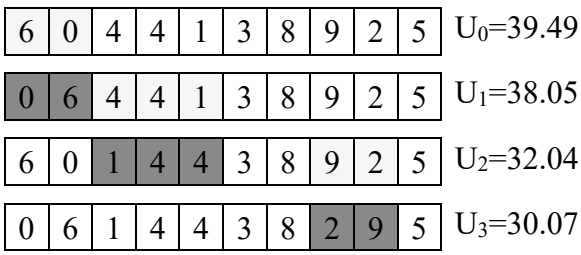

Рис.3. Приклад передобробки із розворотом

Отож, досліджено сім детермінованих алгоритмів передобробки:
– швидка передобробка (QP);
– передобробка з пам'яттю (PM);
– передобробка із розворотом (SR);
– блочна локальна швидка передобробка (BLQP);
– блочна локальна передобробка з пам'яттю (BLPM);
– блочна глобальна швидка передобробка (BGQP);
– блочна глобальна передобробка з пам'яттю (BGPM).

Організація експериментів із дослідження ефективності передобробки

Структури даних. Дослідження відбувалося на масивах цілих 4-байтових чисел. Пам'ять під масиви виділялася безпосередньо засобами C++, створенням std::vector<int> та викликом std::vector<int>::reserve. Сортований масив заповнювався рівномірно розподіленими псевдовипадковими числами від 0 до N-1, де N – кількість елементів масиву x_i. Псевдовипадкові числа були отримані генератором псевдовипадкових величин std::mt19937 з початковим станом, заданим генератором std::random_device. Згенеровані числа приведені до значення із заданого інтервалу за допомогою std::uniform_int_distribution.

Особливості вимірювання часу виконання алгоритму. У зв'язку з нестабільністю роботи потоків операційної системи Windows та технічних засобів, час сортування одного й того ж масиву буде випадковим із деякою дисперсією.

Час сортування визначався як медіанний час із *M*-кратним виконанням сортування одного й того ж масиву даних (дані копіювалися з оригінального невідсортованого масиву перед кожним вимірюванням).

Перед виконанням окремого сортування (паралельного вимірювання) виконувалося очищення кешу. Для цього застосовувався такий спосіб: створювався масив розміром $N = 8 * 2^{20}$ 4-байтових псевдовипадкових чисел, 2 випадкових елементи масиву мінялись місцями за допомогою std::swap N разів.

Технічні засоби. Експерименти виконані на ноутбуці з технічними характеристиками, наведеними у табл. 1.

Таблиця 1

Характеристики технічних засобів

Характеристики центрального процесору	
Поле	Значення
Тип	Intel Core i7-6700HQ, 2600 МГц

Кількість ядер	4
Кеш L1	256 KB per core
Кеш L2	1 MB per core (On-Die, ECC, Full-Speed)
Кеш L3	6 MB (On-Die, ECC, Full-Speed)
Набір інструкцій	64-Bit, Intel® SSE4.1, Intel® SSE4.2, Intel® AVX2, Intel AES
Характеристики оперативної пам'яті	
Поле	Значення
Формфактор	FB-DIMM
Розмір модулю	8 ГБ
Кількість модулів	2 x 8 ГБ

Результати експериментів

Ефективність передобробки будемо оцінювати за показниками (1)-(3). Для дослідження обирались загальновідомі алгоритми сортування, які зазвичай використовують для вирішення задач із різними вимогами. Швидке сортування [9] (Quick sort) є одним із найшвидших універсальних алгоритмів сортування. Використовується для великих списків, коли важлива ефективність. Сортування вставками [10] (Insertion sort) ефективне для малих або частково відсортованих списків. Може використовуватись як базовий варіант для інших алгоритмів. Шейкерне сортування [11] (Cocktail sort) подібне до сортування бульбашкою [12], але виконується в обидва напрямки. Може бути ефективно використане для невеликих або майже відсортованих масивів.

Дослідження невпорядкованості масиву до та після передобробок. Для визначення найбільш ефективного алгоритму попередньої обробки було проведено експеримент, де для масиву випадкових цілих чисел у діапазоні [0, N-1] (де N – розмір масива) вираховувався показник ефективності за формулою (2). Максимально можливе значення ефективності 100% означало б, що результатом передобробки є повністю відсортований масив. На рисунку .4 видно, що найбільшу ефективність має передобробка з пам'яттю. Далі із значенням близько 62% йде швидка передобробка. Передобробка з розворотом показала значення ефективності близьке до 0% для усіх масивів даних з експерименту.

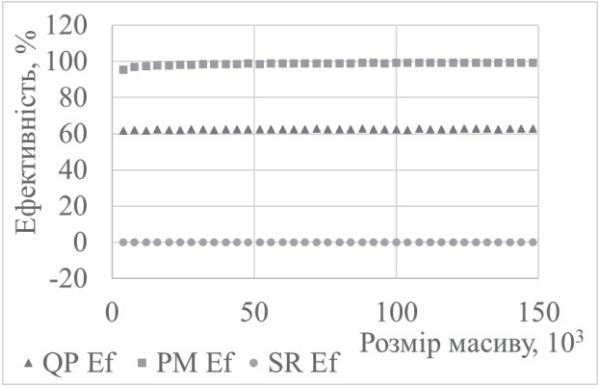

Рис.4. Ефективність передобробок

Дослідження часової ефективності швидкого сортування з використанням різних типів передобробок.

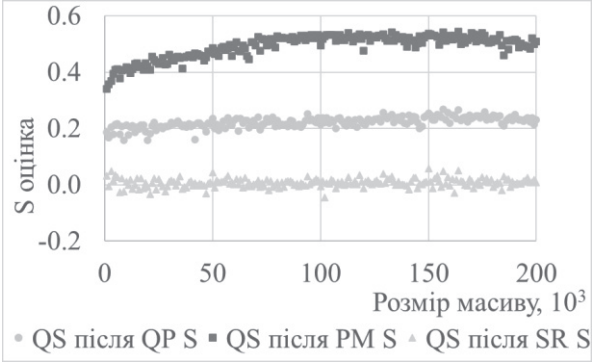

Рис.5. S оцінка швидкого сортування після швидкої, з пам'яттю і передобробки з розворотом

Найбільше зростання часової ефективності швидкого сортування дає попередня обробка з пам'яттю. Значення S оцінки стартує з 0.35 на 1000 елементів і стрімко зростає у разі збільшення розміру масиву. На 50 – 60 тисячах елементів S оцінка досягає 0.5 і лишається на тому ж рівні за подальшого збільшення кількості елементів.

Швидка передобробка має стабільні показники S оцінки у межах 0.18 – 0.25 для усіх розмірів масиву. Передобробка із розворотом має показник близький до 0, плюс мінус 0.05.

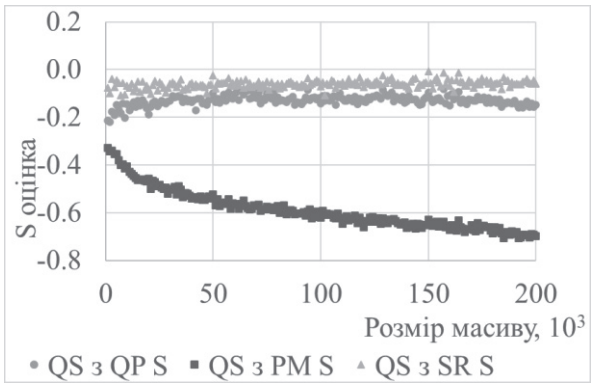

Рис.6. S оцінка швидкого сортування разом із швидкою, з пам'яттю і передобробкою з розворотом

Однак усі досліджені типи передобробок разом зі швидким сортуванням у порівнянні із чистим сортуванням мають негативні значення S оцінки. Для передобробки із розворотом ці значення знаходяться у діапазоні від - 0.02 до - 0.12. Швидка показує значення у діапазоні - 0.1..- 0.2. А для передобробки з пам'яттю S оцінка різко знижується від -0.32, для масиву 1000 елементів, до -0.5 – на 30 тисячах. Далі зі збільшенням розміру масиву продовжується падіння показника до -0.7 на 200 тисячах елементів.

Дослідження впливу блочних передобробок. Блочні локальні передобробки мають позитивний вплив на часову ефективність швидкого сортування. S оцінка сортування після локальних передобробок знаходиться на рівні 0.1 на малих розмірах масиву. На масивах близько ста тисяч і більше елементів показник коливається у межах 0.1 – 0.2.

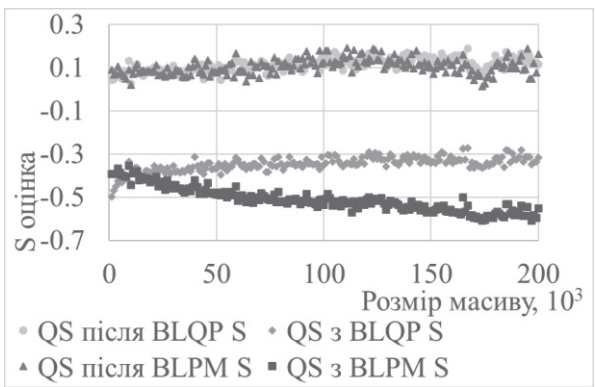

Рис.7. S оцінка швидкого сортування після блочної локальної швидкої та передобробки з пам'яттю

Локальні передобробки разом із швидким сортуванням мають негативні значення S оцінки. Блочна локальна швидка показує значення близько -0.4 – -0.5 на дуже малих розмірах масиву, а після десяти тисяч елементів і до кінці експерименту показує значення у діапазоні -0.3..-0.4. Блочна локальна з пам'яттю починає зі значення близько -0.39 і рівномірно погіршується впродовж усього експерименту, досягаючи значення -0.6.

Блочні глобальні передобробки також мають позитивний вплив на швидке сортування, але дещо менший, ніж локальні. Значення S оцінки швидкої передобробки коливається у межах -0.02 – 0.1 впродовж усього експерименту. Передобробка з пам'яттю показує значення від -0.03 до 0.17.

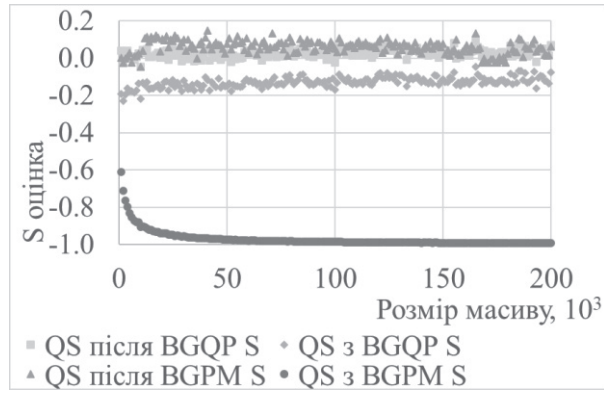

Рис.8. S оцінка швидкого сортування після блочної глобальної швидкої та передобробки з пам'яттю

Значення S оцінки для блочних глобальних передобробок разом із сортуванням є негативним. Швидка передобробка має значення у діапазоні -0.22..-0.07. А з пам'яттю починає з -0.6, сягає -0.9 на десяти тисячах елементів, а після сімдесяти тисяч лишається близьким до -1.

Підсумки. Швидке сортування має складність O(n*log(n)) у кращому випадку, і відсортованість даних у масиві перед початком сортування майже не впливає на час виконання. Усі запропоновані типи передобробок разом зі швидким сортуванням показали довший час виконання, ніж окреме сортування. Однак деякі передобробки показали позитивний вплив на часову ефективність сортування. Передобробка з пам'яттю досягла показника S оцінки 0.5, а швидка передобробка – 0.25. Найкращий показник серед блочних передобробок показала глобальна передобробка з пам'яттю – до 0.17. Решта показала значення на рівні 0.1 – 0.2.

Дослідження часової ефективності сортування вставками з використанням різних типів передобробок. Значення S оцінки для сортування вставками після швидкої передобробки має позитивне значення на малих розмірах масиву. Для масиву довжиною 100 елементів оцінка дорівнює 0.3. Далі значення швидко зростає до 0.5 для масиву довжиною 1000. Зі збільшенням довжини масиву до 100000 елементів, значення S оцінки лишається на рівні 0.55.

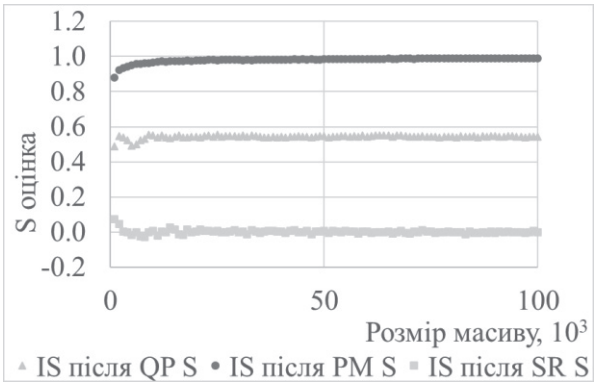

Рис.9. S оцінка сортування вставками після швидкої, з пам'яттю і передобробки з розворотом

Сортування після передобробки з пам'яттю має вищі показники S оцінки. Для масива довжиною 100 елементів оцінка дорівнює 0.55 і стрімко зростає. На 2000 елементів значення S оцінки сягає 0.92, на 10000 – 0.97 й із подальшим збільшенням кількості елементів наближається до 1.

Передобробка із розворотом має показник S оцінки дещо вище 0 на малих розмірах масиву, досягає 0 після 1000 і залишається таким протягом усього експерименту.

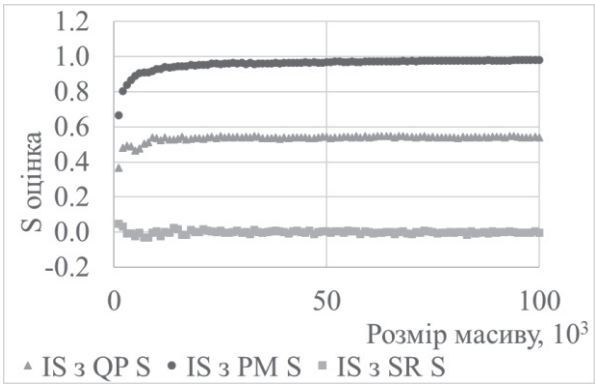

Рис.10. S оцінка сортування вставками зі швидкою, з пам'яттю і передобробкою з розворотом

Швидка передобробка показує негативне значення S оцінки на малих розмірах масиву. Для розміру масиву в діапазоні 200..300 елементів значення близьке до 0. Далі значення рівномірно зростає до 0.53 для масиву довжиною 10000. Подальше збільшення довжини масиву не змінює значення S оцінки.

Передобробка з пам'яттю також має негативне значення S оцінки на малих розмірах масивів. Оцінка для масиву з 300 елементів досягає 0 і продовжує стрімко зростати. На 2000 елементів значення S оцінки сягає 0.8, на 6000 – 0.9 і, за подальшого збільшення кількості елементів, наближається до 1.

Передобробка із розворотом має показник S оцінки дещо нижче 0 на малих розмірах масиву, досягає 0 після 1000 і залишається таким протягом усього експерименту.

Дослідження впливу блочних передобробок. Сортування вставками після обох блочних глобальних передобробок має показник S оцінки у діапазоні від -0.04 до 0.1 на малих розмірах масивів. Із збільшенням довжини масиву діапазон звужується до значень -0.009..0.03. Водночас швидка передобробка має незначну перевагу над передобробкою з пам'яттю.

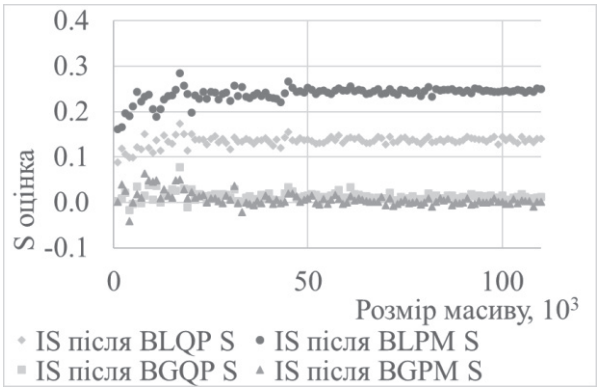

Рис.11. S оцінка сортування вставками після блочних передобробок

Блочні локальні передобробки мають більшу ефективність порівняно з глобальними. Сортування після швидкої передобробки має показник S оцінки 0.029..0.17. Із збільшенням довжини масиву діапазон значень звужується до величин 0.127..0.145.

Блочна локальна передобробка з пам'яттю на малих розмірах масиву показує більший діапазон значень S оцінки 0.128..0.285. За умови збільшення довжини масиву діапазон звужується до значень 0.224..0.271.

Найгірші показники S оцінки показує сортування вставками після блочної глобальної передобробки з пам'яттю. На малих розмірах масивів значення коливаються у діапазоні -0.31..-0.24, для масивів понад 50000 елементів і до кінця експерименту – - 0.29..- 0.28.

Блочна глобальна швидка передобробка дає негативні значення S оцінки на масивах малої довжини. Досягає 0 на масивах довжиною 5000 елементів і лишається на тому ж рівні до кінця експерименту.

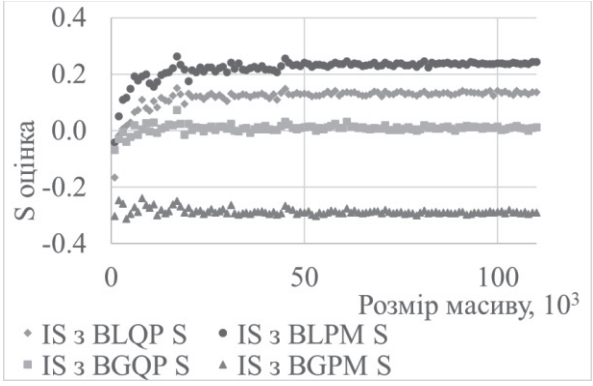

Рис.12. S оцінка сортування вставками разом із блочними передобробками

Блочні локальні передобробки показують вищий рівень значень S оцінки ніж глобальні. Швидка передобробка має негативне значення на малих розмірах масивів. Для масивів довжиною 15000 і до кінця експерименту S оцінка лишається на рівні 0.11..0.13.

Передобробка з пам'яттю також має негативні значення S оцінки на малих розмірах масиву. На масивах довжиною понад 15000 показує значення у діапазоні 0.2..0.26.

Підсумки. Сортування вставками має складність O(n) у кращому випадку – на повністю відсортованому масиві. Тому передобробки разом із сортуванням мають переважно позитивні значення S оцінки. А через те, що час передобробки дуже малий порівняно із часом сортування, результати S-оцінок для алгоритму сортування після і разом з передобробкою відрізняються на 0.01 – 0.02. Передобробка з розворотом має показники на рівні 0, швидка передобробка – 0.53, передобробка з пам'яттю – наближається до 1 зі збільшенням розміру масиву. Серед блочних найбільші показники у локальної передобробки з пам'яттю – 0.26, локальна швидка має – 0.13, глобальна швидка – 0. Найнижчі показники у глобальної передобробки з пам'яттю – -0.

Дослідження часової ефективності шейкерного сортування з використанням різних типів передобробок. Шейкерне сортування після швидкої передобробки має позитивне значення S оцінки на всіх довжинах масивів з експерименту. На малому розмірі масиву S оцінка дорівнює 0.54 і швидко зростає зі збільшенням розміру масиву. На масивах близько 1000 елементів сягає 0.58, а вже після 7000 елементів виходить на свій постійний рівень – 0.7.

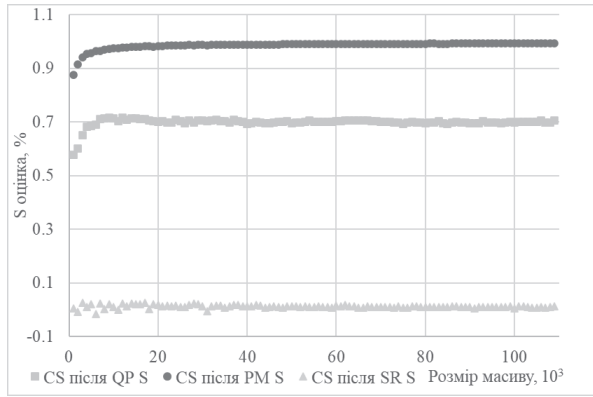

Рис.13. S оцінка шейкерного сортування після швидкої, з пам'яттю і передобробки з розворотом

Передобробка з пам'яттю має досить велике значення (0.7) S оцінки на малих розмірах масиву. Масив з 1000 елементів досягає 0.87, а після 10000 – 0.98. Далі на всіх довжинах масивів, використаних в експерименті, значення S оцінки сягає і перевищує 0.99.

Шейкерне сортування після передобробки з розворотом на масивах довжиною до 1000 показує значення S оцінки у діапазоні -0.02..0.06. На довших масивах діапазон значень звужується і лишається рівним 0.0..0.02.

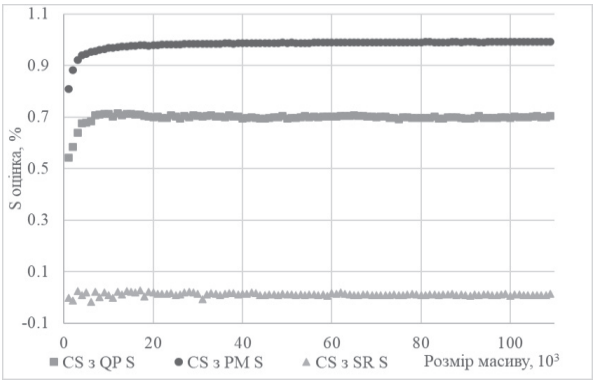

Рис.14. S оцінка шейкерного сортування разом із швидкою, з пам'яттю і передобробкою з розворотом

Усі вищезгадані типи передобробок займають мало часу порівняно з шейкерним сортуванням. Тому відмінність у показниках S оцінки можна побачити на малих розмірах масивів. Початкове значення для швидкої передобробки для масиву довжиною 100 дорівнює 0.3, а вже для 1000 досягає 0.54, після 7000 елементів і до кінця експерименту – 0.7.

Передобробка з пам'яттю має досить мале значення 0.15 для масиву зі 100 елементів. А при довжині рівній 1000 – 0.81. На великих розмірах сягає 0.99 і наближається до 1.

Значення для передобробки з розворотом знаходяться у діапазоні -0.034..0.008 для масивів довжиною до 1000 елементів. Далі усі значення S оцінок сортування з передобробками співпадають із результатами сортування після передобробок.

Дослідження впливу блочних передобробок. Найкращі показники S оцінки на великих розмірах масивів показує блочна локальна передобробка з пам'яттю. На масивах до 1000 елементів значення збільшуються від 0.16 до 0.29. На 10000 дорівнює 0.54, а на 20000 досягає 0.6 і лишається на тому ж рівні до кінця експерименту.

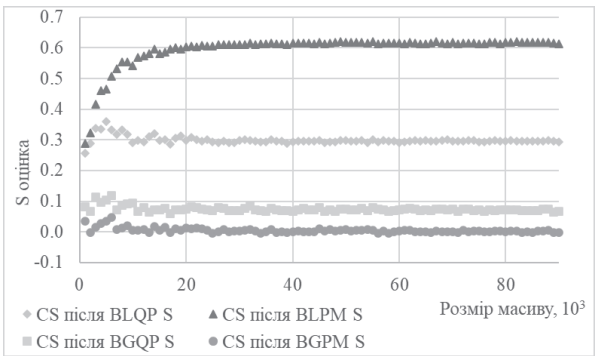

Рис.15. S оцінка шейкерного сортування після блочних передобробок

Блочна локальна швидка передобробка має показники S оцінки у діапазоні 0.12..0.26 на масивах довжиною до 1000 елементів. Для масивів 1000–10000 елементів діапазон розширюється і має значення 0.26..0.34. Для більших масивів показник лишається на рівні 0.3 протягом усього експерименту.

Блочні глобальні передобробки показують значно нижні результати S оцінки порівняно з локальними. Швидка передобробка має значення у діапазоні 0.01..0.13 на масивах довжиною до 10000. На довших масивах показники варіюються від 0.07 до 0.08.

Блочна глобальна передобробка з пам'яттю має найнижчі показники S оцінки. На масивах до 10000 елементів – -0.04..0.06, на довших масивах – -0.06..0.04. Зі збільшенням довжини масиву наближається до нуля.

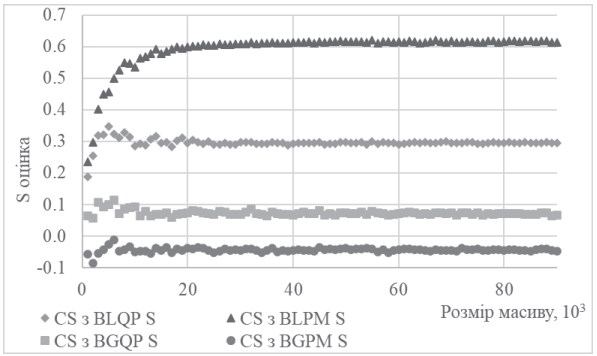

Рис.16. S оцінка шейкерного сортування разом із блочними передобробками

Шейкерне сортування разом із блочними локальними передобробками має нижчі показники S оцінок, порівняно із сортуванням після передобробок, на малих розмірах масивів. Для швидкої передобробки на масивах довжиною менше 10000 елементів значення оцінки стрімко зростають від -0.26 до 0.19. А для передобробки з пам'яттю – від -0.29 до 0.24. Далі протягом усього експерименту значення S оцінок співпадають із результатами сортування після блочних локальних передобробок.

Блочна глобальна швидка передобробка на масивах довжиною до 10000 елементів має показник S оцінки, що зростає з -0.07 до 0.06. Для довших масивів значення знаходяться у діапазоні 0.065..0.076.

Блочна глобальна передобробка з пам'яттю має негативні показники S-оцінки протягом усього експерименту. На масивах довжиною до 10000 елементів значення лежать у діапазоні -0.29..-0.05, а більше 10000 – -0.06..-0.01.

Підсумки. Шейкерне сортування, як і сортування вставками, має складність O(n) у кращому випадку, що підтверджують високі експериментально отримані показники S оцінки для сортування разом із передобробками. Для швидкої передобробки – 0.7, передобробки з пам'яттю – 0.99. Ефективність передобробки із розворотом близька до 0. Серед блочних алгоритмів найбільш ефективна локальна з пам'яттю – 0.6, локальна швидка – 0.3, глобальна швидка – 0.08, глобальна з пам'яттю – близька до 0.

Висновки

Встановлено, що із розмежуванням процесів обробки та сортування можливо отримати значне скорочення часу виконання алгоритмів сортування. Більш ефективна передобробка для алгоритмів сортування, що прискорюються з майже відсортованими даними.

Передобробка з пам'яттю дозволяє отримати краще впорядковані масиви даних, але потребує більше часу та пам'яті порівняно зі швидкою передобробкою.

Передобробка із розворотом не погіршує, але і не поліпшує в межах похибки час сортування для усіх досліджених алгоритмів.

Для великих обсягів даних, більших, ніж довжина кешу, ефективнішим може стати використання блочних локальних або глобальних передобробок. Через їхню особливість – не виконувати перестановки, якщо передбачений елемент знаходиться за межами блоку, - можна мінімізувати або зовсім уникнути промахів кешу (cache miss).

У випадках, коли сумарний час передобробки і сортування перевищує час сортування, використання передобробки видається недоцільним. Однак залежно від архітектури і вимог до даних у програмі передобробка може викликатись один або багато разів протягом роботи програми, за умови надходження певної кількості нових даних. Таким чином дані можна тримати частково відсортованими. А за потреби у повністю відсортованих даних, викликати безпосередньо алгоритм сортування, який виконається швидше через те, що дані знаходяться у «передобробленому» стані.

Найбільший ефект застосування запропонованих алгоритмів передобробки виявлено під час застосування передобробки з пам'яттю разом із сортуванням вставками і шейкерним сортуванням. Значення S оцінки у цих випадках наближається до 1. Також досить високу ефективність показує швидка передобробка для вищезгаданих алгоритмів сортування. Серед блочних передобробок локальні передобробки показали більшу ефективність, ніж глобальні.

Жодний тип передобробки не мав позитивного впливу у використанні разом зі швидким сортуванням. Час виконання передобробки перевищував виграш від скорочення часу сортування переднеоброблених даних.

Література

1. Knuth, Donald. The Art Of Computer Programming, vol. 3: Sorting And Searching. : Addison-Wesley, 1973.
2. Timsort [Online] – Available from: https://svn.python.org/projects/python/trunk/Objects/listsort.txt, last accessed 2023/08/07.
3. Musser, David R. (1997). "Introspective Sorting and Selection Algorithms". Software: Practice and Experience. 27 (8): 983–993.
4. Adnan Saher Mohammed, Şahin Emrah Amrahov, Fatih V. Çelebi. Bidirectional Conditional Insertion Sort algorithm; An efficient progress on the classical insertion sort. Future Generation Computer Systems, Volume 71, June 2017, 102-112.
5. Shrinu Kushagra, Alejandro López-Ortiz, Aurick Qiao, J. Ian Munro. Multi-Pivot Quicksort: Theory and Experiments. 2014 Proceedings of the Meeting on Algorithm Engineering and Experiments (ALENEX), 47-60
6. Шинкаренко В.І., Дорошенко А.Ю., Яценко О.А., Разносілін В.В., Галанін К.К. Двокомпонентні алгоритми сортування Проблеми програмування. – 2022. – № 3-4. – С. 32-41. – Бібліогр.: 18 назв. – укр.

7. Abbas Mubarak, Sajid Iqbal, Qaisar Rasool, Nabeel Asghar, Neetu Faujdar, & Abdul Rauf. (2022). Preprocessing: A method For Reducing Time Complexity. Journal of Computing & Biomedical Informatics, 4(01), 104–117.
8. Шинкаренко В.І Сравнительный анализ временной эффективности функционально эквивалентных алгоритмов / В. И. Шинкаренко // Проблемы программирования. – 2001. – № 3-4. – С. 31-39
9. Hoare, C. A. R. (1962). "Quicksort". Comput. J. 5 (1): 10–16.
10. Insertion sort [Online] – Available from: https://en.wikipedia.org/wiki/Insertion_sort, last accessed 2023/08/07.
11. Knuth, Donald E. (1973). "Sorting by Exchanging". Art of Computer Programming. Vol. 3. Sorting and Searching (1st ed.). Addison-Wesley. pp. 110–111.
12. Bubble sort [Online] – Available from: https://en.wikipedia.org/wiki/Bubble_sort, last accessed 2023/08/07.

Одержано: 18.10.2023

Про авторів:

Шинкаренко Віктор Іванович,
доктор технічних наук, професор,
Кількість публікацій в українських виданнях – більше 200
Кількість зарубіжних публікацій – більше 30
індекс Хірша – 6
https://orcid.org/0000-0001-8738-7225
E-mail: shinkarenko_vi@ua.fm

Макаров Олексій Вікторович,
аспірант,
Кількість публікацій в українських виданнях – 1
https://orcid.org/0009-0003-0921-155X
E-mail: makarovov@hotmail.com

Місце роботи авторів:

Український державний університет науки і технологій,
49010, Україна, Дніпро,
вул. академіка Лазаряна, 2.
E-mail: office@ust.edu.ua

УДК 004.65 http://doi.org/10.15407/pp2023.04.015

Г.Ю. Проскудіна, К.О. Кудім, В.А. Резніченко

VUFIND: ВІДКРИТЕ РІШЕННЯ ДЛЯ ІНТЕГРАЦІЇ БІБЛІОТЕЧНИХ КОЛЕКЦІЙ

У статті розглядається система VuFind як відкрите рішення для ефективної інтеграції бібліотечних колекцій. VuFind є потужним пошуковим інтерфейсом, розробленим для поліпшення доступу до різноманітних ресурсів, включно з книжками, статтями, журналами, науковими звітами, іншими матеріалами. Автори обговорюють ключові особливості VuFind, такі як гнучкість налаштувань, можливості пошуку, підтримка метаданих та інтеграція з різними джерелами даних. Наголошується на ролі VuFind у спрощенні пошуку для користувачів та оптимізації управління колекціями з різних бібліотек. VuFind надає відкрите та доступне рішення для створення сучасних бібліотечних систем, сприяючи ефективній інтеграції та підвищенню задоволеності користувачів.

Ключові слова: інтеграція, збір даних, харвестер, протокол OAI-PMH, простий пошук, розширений пошук

Вступ

Сучасні бібліотеки стикаються зі зростаючим попитом на зручний та ефективний доступ до різноманітних ресурсів, починаючи від традиційних книжок до електронних журналів і цифрових архівів. В умовах цього динамічного інформаційного ландшафту ключовим фактором є здатність ефективно управляти та інтегрувати різні елементи бібліотечних колекцій.

У цьому контексті VuFind виокремлюється як потужне та відкрите рішення, призначене для покращення доступу до бібліотечних ресурсів. VuFind – це пошуковий інтерфейс із відкритим кодом, розроблений для забезпечення інтеграції та ефективного управління бібліотечними колекціями. Гнучкість і багатофункціональність роблять його привабливим для різних бібліотек: від академічних установ до громадських бібліотек і спеціалізованих колекцій.

Однією з ключових рис VuFind є його здатність об'єднувати результати пошуку з різних джерел даних, забезпечуючи користувачам єдиний і зручний інтерфейс. Пошук у VuFind дає змогу ефективно шукати одразу в декількох каталогах бібліотек, цифрових архівах, базах даних та інших ресурсах, надаючи комплексний огляд доступних матеріалів.

Ґрунтуючись на відкритому вихідному коді, VuFind забезпечує бібліотеки гнучкістю в налаштуванні та адаптації під свої унікальні потреби. Розширені можливості конфігурації дають змогу впроваджувати індивідуальні налаштування відповідно до вимог кожної організації.

В рамках виконання частини проєкту НАНУ "Відкрита наука" перед нами було поставлено завдання створення системи інтеграції або харвестера ресурсів із різних відкритих академічних джерел з метою отримання зручного і потужного інструменту пошуку та доступу до інформації для користувачів.

Таке завдання ми вирішуємо вже не вперше. Так, у роботі [1] були описані підходи до здійснення інтеграції ресурсів з різних джерел та був запропонований варіант її практичної реалізації. Зокрема, було розглянуто клас систем інтеграції, в яких за основу взято технологію Ініціативи відкритих архівів (Open Archive Initiative – OAI),[1] де всі учасники забезпечують інтерфейс OAI і використовують протокол OAI-PMH або для надання свого вмісту (провайдери даних), або для збору цього вмісту (провайдери сервісів, харвестери).

У процесі виконання проєкту був здійснений аналіз сучасних програмних продуктів, які використовуються для створення харвестера. І на основі цього аналізу вибір припав на систему VuFind. Вона була розроблена в університеті Вілланова, США [2], перша її версія вийшла 2010 року, а в липні 2023 була випущена 9-та версія. Це відкрите програмне забезпечення

[1] https://www.openarchives.org/pmh/

поширюється за стандартною публічною ліцензією GNU[2].

Як правило перед харвестером стоять три основні мети:

1. Збір та інтеграція метаданих з різних джерел електронних ресурсів;
2. Організація пошуку і видачі відповідних ресурсів;
3. Передача метаданих з власного харвестеру іншим харвестерам.

Усі ці три функції можна виконати, взявши за основу систему VuFind.

Що ж до реалізації харвестера, систему VuFind було встановлено і протестовано його можливості підключенням 5 електронних бібліотек, з яких загалом було зібрано приблизно 200 тис. статей. Наразі триває робота з перевірки та уточнення форматів даних, що передаються, перевірки та налагодження передачі і збору даних з таких бібліотечних та журнальних систем як DSpace, Eprints, OJS [3-5].

У VuFind механізм пошуку реалізований на пошуковій системі Apache Solr, яка в свою чергу заснована на дослідницькій бібліотеці Lucene, яку на сьогодні використовують більшість електронних бібліотечних систем. Наприклад, Наукова бібліотека періодичних видань НАНУ [3], яка функціонує з 2007 року, про пошукові можливості якої йдеться у роботі [6] і в ряді інструкцій.

Було також перевірено, яким чином цей харвестер передає метадані іншим харвестерам, скориставшись зокрема, системою пошуку Відкритих архівів України[3].

Слід також зазначити, що VuFind має документацію з усіх аспектів проєкту: розгортання, налаштування, завантаження даних та їх індексації, адміністрування, подальшої модифікації та розширення. Сайт демонстрації VuFind[4] наживо показує вигляд і функції системи. Крім того, документація використовує технологію Wiki[5], де користувачі VuFind розміщують практичні поради, сценарії та іншу корисну інформацію.

Ця стаття розповідає про наш досвід вивчення та впровадження цієї системи. Також розкриваються основні характеристики VuFind, включно з його можливостями, перевагами, а також викликами та перспективами використання.

1. Особливості VuFind

Слова "пошук" (search), "виявлення" (discover) і "спільне використання" (share) на логотипі системи VuFind (рис. 1) очевидно відображають основні концепції та цілі системи. Розглянемо ці поняття докладніше:

Рис. 1. Логотип системи VuFind

Пошук (Search). Пошук у системі VuFind – це основна функція, що надається користувачам. Він дозволяє користувачам знаходити інформацію в бібліотечних колекціях і ресурсах. Включає в себе пошук книг, журналів, статей, електронних ресурсів, цифрових архівів і багатьох інших видів даних.

Пошук здійснюється з використанням інтуїтивних інтерфейсів, що робить його доступним і зручним для користувачів усіх рівнів. Користувачі можуть виконувати пошук за ключовими словами, авторами, темами та іншими критеріями, що допомагає їм знайти потрібні ресурси швидко і точно.

Виявлення (Discover). Виявлення у VuFind означає, що система надає не тільки результати пошуку, а й активно допомагає користувачам відкривати нові та цікаві для них ресурси. Це може включати рекомендаційні системи, пов'язані ресурси, аналоги тощо.

Наприклад, система може запропонувати користувачеві пов'язані статті або книжки, засновані на його запиті, що сприяє глибшому дослідженню теми.

Спільне використання (Share). Спільне використання у VuFind дає змогу користувачам легко ділитися знайденими

[2] https://en.wikipedia.org/wiki/GNU_General_Public_License
https://oai.org.ua/

[4] http://vufind.org/demo/
[5] https://vufind.org/wiki/

ресурсами з іншими. Це може містити функції спільного використання через соціальні медіа, надсилання посилань електронною поштою, створення закладок і багато іншого.

Можливості спільного використання допомагають користувачам поширювати інформацію та ресурси між колегами, друзями та соціальними мережами, що сприяє ширшому доступу до знань та інформації.

Загалом ці три поняття (пошук, виявлення та спільне використання) відображають ключові аспекти того, як система VuFind допомагає користувачам досліджувати, знаходити й ділитися інформацією та ресурсами в бібліотечних колекціях та інших джерелах даних.

Можна також виділити наступне:

Безліч джерел даних. VuFind підтримує різноманітні джерела даних, включно з різноманітними системами управління бібліотеками, електронними каталогами, репозиторіями, базами даних і багатьма іншими. Це дає змогу інтегрувати та керувати різними видами ресурсів в одній системі.

Персоналізація та профілі користувачів. Система уможливлює налаштування профілів користувачів, що дає змогу адаптувати інформацію та ресурси під конкретні потреби користувачів. Це містить в собі налаштування рекомендаційних систем, оформлення інтерфейсу та інші функції, що сприяють кращій персоналізації.

Інтеграція з відкритими інтернет-сервісами. VuFind забезпечує інтеграцію з різними відкритими інтернет-сервісами, такими як Google Books, OpenStreetMap тощо. Це розширює можливості пошуку та надає додаткові відомості про ресурси.

Багато локацій і багатокористувацький доступ. VuFind підтримує роботу з великою кількістю бібліотечних систем, а також забезпечує багатокористувацький доступ. Це дає змогу організаціям із розподіленою структурою ефективно керувати своїми ресурсами та надавати широкий доступ користувачам.

Відкрите програмне забезпечення і співтовариство. Оскільки VuFind розробляється як відкрите програмне забезпечення, це дає змогу адаптувати і розширювати систему відповідно до своїх потреб. Існує також активне товариство розробників і користувачів, які співпрацюють для поліпшення системи.

2. Харвестер даних

VuFind може використовуватися як харвестер даних для збору інформації з різних джерел та інтеграції її в єдину інтегровану систему. Ось декілька прикладів використання VuFind як харвестера даних:

Імпорт каталогів бібліотечних систем. VuFind може використовуватися для імпорту даних з наявних бібліотечних систем або каталогів. Бібліотека може використовувати VuFind для збору даних з її інтегрованих бібліотечних систем (Integrated Library System – ILS) і надавати своїм користувачам більш зручний та сучасний інтерфейс для пошуку та навігації.

Інтеграція з цифровими архівами та репозиторіями. VuFind може інтегрувати цифрові архіви та репозиторії, що дає змогу надавати доступ до цифрових колекцій, а також ресурсів, як-от електронні тезауруси та архіви з відкритим доступом.

Інтеграція із зовнішніми базами даних та онлайн-ресурсами є одним із сильних аспектів системи VuFind**.** Система може бути налаштована для збору даних із різноманітних зовнішніх баз даних та онлайн-ресурсів, таких як наукові журнали, бази даних, електронні книги та інші джерела. Це уможливлює для користувачів пошук інформації в різних ресурсах і різних типах через єдиний інтерфейс і отримувати результати пошуку, а також посилання на повні тексти статей і журналів.

Деякі бібліотеки інтегрують відкриті ресурси, як-от відкриті підручники та онлайн-курси, у свій каталог з використанням VuFind. Це дає змогу студентам і викладачам знаходити та використовувати безкоштовні освітні матеріали.

Також можлива інтеграція з музейними колекціями. Деякі музеї інтегрують свої колекції з системою VuFind. Відвідувачі можуть шукати і виявляти твори мистецтва та артефакти, що знаходяться в музейних колекціях, через бібліотечний інтерфейс.

Інтеграція з геопросторовими даними. У разі, якщо бібліотеки та установи,

пов'язані із геопросторовою інформацією, VuFind може інтегрувати геопросторові бази даних і картографічні ресурси, що дає змогу користувачам шукати і взаємодіяти з географічною інформацією (рис. 2).

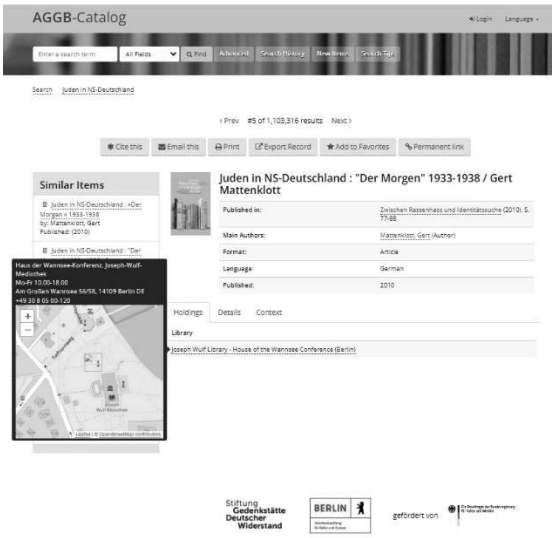

Рис. 2. Сторінка інтегрованої бібліотечної системи науково-дослідних інститутів Німеччини AGGB-Catalog[6]

Ці приклади демонструють гнучкість і потужність системи VuFind під час інтеграції з різноманітними зовнішніми базами даних та онлайн-ресурсами включно з бібліотечними каталогами, цифровими репозиторіями, архівами та музейними колекціями.

Імпорт і аналіз даних відкритого доступу. VuFind може використовуватися для збору даних із джерел відкритого доступу, таких як відкриті архіви, бібліотеки з відкритими даними та онлайн-репозиторії. Це дає змогу бібліотекам та установам збирати та інтегрувати безкоштовні ресурси.

Багато бібліотек прагнуть інтегрувати відкриті електронні книги та журнали, доступні у відкритому доступі, у свої колекції. З використанням VuFind, бібліотека може налаштувати процес імпорту даних, включно з метаданими та доступними файлами, з різних відкритих джерел, таких як Project Gutenberg або відкриті репозиторії наукових публікацій. Багато університетів та організацій підтримують відкриті архіви та репозиторії, що містять наукові роботи, дисертації та інші матеріали. З використанням VuFind бібліотеки можуть інтегрувати ці ресурси, забезпечуючи доступ до актуальних наукових публікацій доступних і безкоштовних для всіх дослідників і студентів.

Загалом VuFind надає гнучку та потужну інфраструктуру для збору, інтеграції та надання доступу до різних видів даних та інформації, що робить його корисним інструментом для бібліотек та інформаційних установ, які прагнуть збагатити свої колекції та забезпечити легкий доступ до інформації користувачам.

3. Приклад застосування

VuFind застосовують здебільшого у бібліотечних системах для полегшення пошуку та доступу до ресурсів бібліотек. На офіційному сайті VuFind[7] зібрана вся інформація про діючі системи на основі VuFind.

Її потужні можливості яскраво демонструє система BASE (Bielefeld Academic Search Engine)[8]. Це одна з найбільш об'ємних пошукових систем у світі, особливо для академічних веб-ресурсів таких як наукові статті, дисертації, препринти тощо. [7]. BASE надає понад 340 мільйонів документів від більш ніж 11 000 постачальників контенту (рис. 3). Тут можна отримати доступ до повних текстів близько 60% безкоштовних (у відкритому доступі) проіндексованих документів BASE знаходиться у фондах бібліотеки Білефельдського університету. На сайті ведеться статистика[9] про те, як поповнювалась бібліотека постачальниками контенту та документами за майже 20 років існування цієї системи.

Особливо цікавим для нас є розроблений перелік вимог BASE до своїх постачальників контенту "Золоті правила менеджерів ропозиторія" (Golden Rules for Repository Managers[10]). Розглянемо деякі з них:

[6] https://neu.aggb-katalog.de/vufind/

[7] https://vufind.org/wiki/community:installations

[8] https://www.base-search.net/about/en/index.php

[9] https://www.base-search.net/about/en/about_statistics.php

[10] https://www.base-search.net/about/en/faq_oai.php

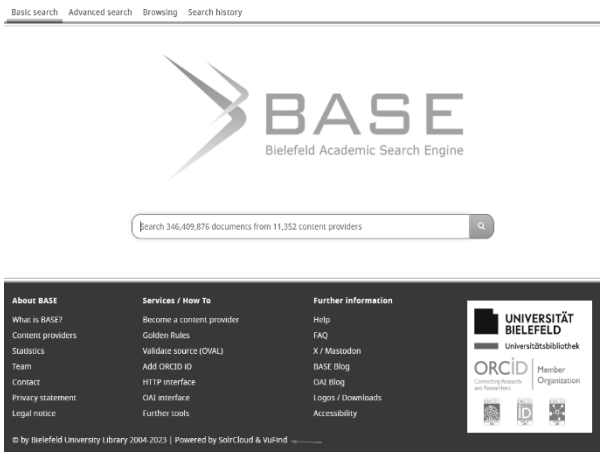

Рис. 3. Домашня сторінка BASE

– Індексація метаданих здійснюється для всіх видів академічних ресурсів, які забезпечують інтерфейс OAI і використовують OAI-PMH для надання свого вмісту. Проіндексовані дані зберігаються на серверах Білефельдського університету.

– Для перевірки відповідності репозиторію вимогам BASE можна скористатись валідатором OVAL[11] або OAI-PMH Validator[12].

– Якщо джерело не має інтерфейсу OAI, пряме індексування такого джерела наразі неможливе. У цьому випадку слід завантажити документи в агрегатори, які вже проіндексовані в BASE або зареєструвати джерело відкритого доступу в DOAJ[13]. Ці постачальники контенту регулярно індексуються в BASE.

– OAI інтерфейс провайдера контенту вільно доступний, стабільний і постійно відповідає запиту ListRecords у форматі oai_dc, повертає результати без тайм-ауту чи помилки виводу. Слід регулярно перевіряти функціональність інтерфейсу OAI, зокрема, за допомогою браузера.

– У полі *Ідентифікувати дані вашого інтерфейсу OAI* у полі adminEmail вказується адреса e-mail, за якою можна зв'язатися з технічним оператором інтерфейсу OAI. Ця адреса доступна на домашній сторінці, що гарантує прямий контакт з оператором контент-провайдера.

Тут наведені вимоги, що стосуються лише інтерфейсу OAI, але на сайті BASE є й інші: про поступове збирання записів (incremental harvesting), тобто щоразу у харвестер завантажуються не всі записи постачальника контенту, а лише нові або ті, в яких були зміни; ідентифікацію змін метаданих окремих записів; видалення записів; інформацію про принципові зміни (зміни назви постачальника або URL-адреси інтерфейсу OAI); вимоги до полів метаданих DC (рис. 4) з відповідними роз'ясненнями, посиланнями та прикладами, з помітками обов'язковості даних (повинно бути, має бути, може бути).

Інформація	Елемент в oai_dc	Критерії
URL видання	<dc: ідентифікатор>	Повинно бути
Назва	<dc:назва>	Має бути
Автор	<dc:creator>	Має бути
Вид публікації	<dc:тип>	Має бути
Дата публікації	<dc:дата>	Має бути
Мова документа	<dc:мова>	Має бути
Права доступу та повторного використання	<dc:права>	Має бути
Посилання / Цитата	<dc: джерело>	Має бути
Інші сторони, причетні до видання	<dc:contributor>	Може бути
Формат файлу	<dc:формат>	Може бути
опис	<dc:опис>	Може бути
Ключові слова	<dc:subject>	Може бути
Видавець	<dc:publisher>	Може бути
Пов'язані документи	<dc:відношення>	Може бути
Розмежування змісту	<dc:coverage>	Може бути

Рис. 4. Примітки щодо окремих полів метаданих у BASE

У майбутньому планується розробити подібні правила і вимоги до постачальників контенту нашого харвестера.

4. Опис системи VuFind

4.1. Архітектура системи VuFind складається з кількох ключових компонентів (деякі з яких показані на рис. 5):

1. Інтерфейс користувача (UI). Це компонент, який надає користувачу інтерактивний інтерфейс для виконання пошуку та навігації бібліотечними ресурсами. UI зазвичай містить веб-інтерфейс, де користувачі можуть вводити запити й отримувати результати пошуку.

2. Ядро системи (Application Core). Цей компонент є основою системи VuFind і відповідає за обробку запитів користувача, а також координацію роботи інших компонентів. Ядро керує пошуком, аналізом запитів, а також інтеграцією даних з різних джерел.

3. Індексація та пошуковий движок (Indexing and Search Engine, Apache SOLR). Цей компонент відповідає за індексацію та

[11] http://oval.base-search.net/
[12] https://validator.oaipmh.com/
[13] https://doaj.org/application/new

пошук бібліотечних даних. Він створює пошуковий індекс, який прискорює процес пошуку, і виконує його на основі запитів користувачів.

4. Інтеграція з джерелами даних (Data Source Integration). VuFind інтегрує дані з різних джерел, як-от ILS, цифрові репозиторії, бази даних та інші. Цей компонент відповідає за збір та інтеграцію метаданих і ресурсів із цих джерел.

5. Управління авторизацією та автентифікацією (Authentication and Authorization): Для забезпечення доступу до деяких ресурсів і функцій системи, VuFind містить компонент управління авторизацією та автентифікацією користувачів. Це дозволяє обмежувати доступ до конфіденційних даних і функцій тільки авторизованим користувачам.

6. Пошукові фільтри та розширення (Search Filters and Extensions): Цей компонент уможливлює налаштування та розширення функціональності пошуку. Тут можна додавати додаткові фільтри, а також розширювати можливості пошуку відповідно до вимог бібліотеки.

7. База даних (Database, DB). Внутрішня база даних використовується для зберігання метаданих та іншої інформації, необхідної для функціонування системи.

8. Кешування та керування ресурсами (Caching and Resource Management): Цей компонент керує кешуванням даних і ресурсів, що допомагає прискорити обробку запитів і зменшити навантаження на сервер.

9. Інтерфейс для адміністрування (Admin Interface): Для адміністраторів системи надається веб-інтерфейс, за допомогою якого можна налаштовувати і керувати системою. Розробники та адміністратори можуть налаштовувати та розширювати її, щоб відповідати специфічним потребам своєї бібліотеки чи установи.

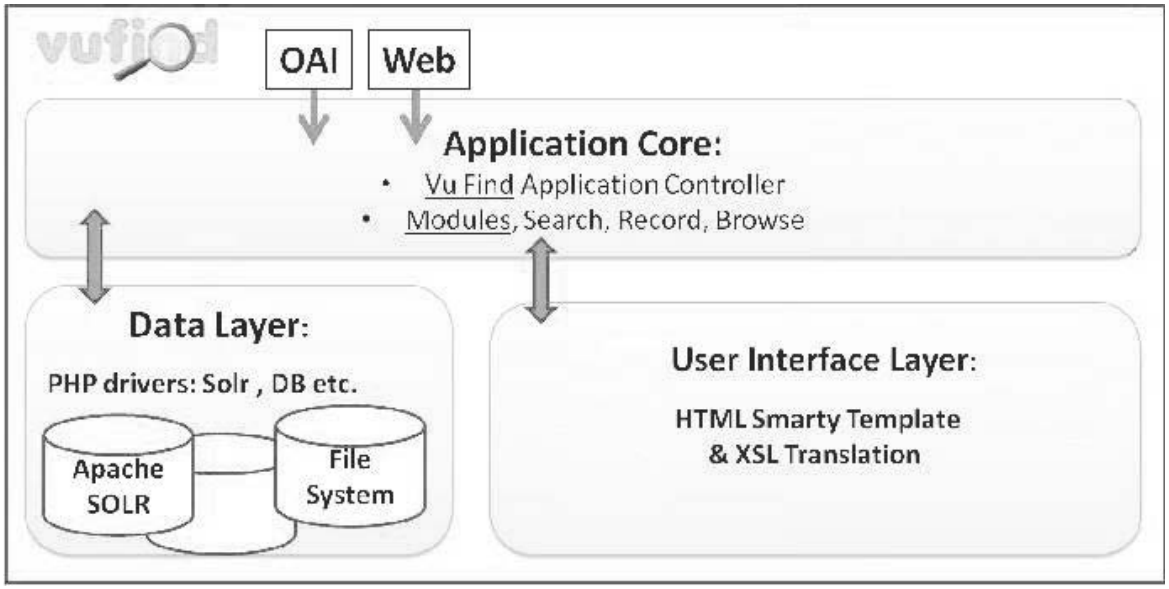

Рис. 5. Архітектура VuFind

4.2. Система користувачів у VuFind дає змогу створювати та керувати різними типами облікових записів, які мають різні права та можливості в системі. У VuFind виділяють три основні групи користувачів:

Анонімні користувачі – це ті, котрі не увійшли в систему за допомогою облікового запису. Їм надаються всі можливості пошуку: вони можуть здійснювати перегляд списків та описів різних документів, робити пошук і фільтрацію, а також використовувати деякі сервіси, такі як RSS-стрічки, експорт даних і надсилання посилань електронною поштою. Однак вони не можуть зберігати свої пошукові запити та їх результати, створювати свої списки документів, залишати відгуки або резервувати документи.

Зареєстровані користувачі – це ті, котрі створили собі обліковий запис у VuFind або увійшли за допомогою обліково-

го запису з іншої системи, наприклад, LDAP[14] або Shibboleth[15]. Вони можуть робити все те ж саме, що й анонімні користувачі, а також їм надаються додаткові функції:

– Переглядати історію своїх пошукових запитів і результатів, а також зберігати їх під своїм обліковим записом.

– Створювати і редагувати списки вибраних документів, ділитися ними з іншими користувачами.

– Додавати або обирати власні теги для каталогізації чи пошуку ресурсів у системі VuFind.

– Залишати відгуки, коментарі та оцінки до документів, а також читати відгуки інших користувачів.

– Підписуватися на RSS-канали для отримання новин про нові документи за темами, що цікавлять.

– Використовувати різні інструменти для роботи з документами, як-от експорт у формати BibTeX, EndNote, RefWorks, Zotero тощо, надсилання електронною поштою, друк, цитування тощо.

Адміністратори - це ті, хто мають спеціальні привілеї для управління системою VuFind. Вони можуть робити все те ж саме, що й зареєстровані користувачі, а також налаштовувати параметри системи, імпортувати та експортувати дані, переглядати статистику та журнали, керувати користувачами та групами користувачів, одержувати і редагувати відгуки тощо.

Для створення та управління користувачами у VuFind використовується спеціальний модуль UserAccounts, який дозволяє налаштовувати різні опції для різних типів користувачів, як-от: способи автентифікації, ролі та права доступу, поля профілю тощо. Модуль UserAccounts також підтримує інтеграцію з іншими модулями VuFind, такими як Social, Favorites, Feedback тощо, які розширюють функціональність системи для користувачів.

4.3. Підтримка різних форматів метаданих. Набір пошукових полів або характеристик для документів VuFind відповідає стандарту Dublin Core Metadata Initiative[16] (DCMI), який є міжнародним стандартом для опису ресурсів у мережі веб. DCMI визначає 15 характеристик (елементів метаданих), зокрема, заголовок, автор, дата, формат, ідентифікатор тощо, які можуть бути використані для опису будь-якого типу ресурсу, включно з книгами, журналами, зображеннями, відео тощо. DCMI також надає набір схем кодування, словників і онтологій для розширення й уточнення значень елементів метаданих. VuFind використовує DCMI для створення єдиного і сумісного формату опису документів, які зберігаються в різних бібліотечних системах і базах даних. VuFind також підтримує пошук за метаданими з використанням стандарту SPARQL, що є мовою запитів до даних, представлених у форматі RDF, який є основним форматом даних у семантичному вебі.

Окрім стандарту Dublin Core у VuFind можна використовувати інші формати метаданих. VuFind підтримує різні формати метаданих, такі як MARC 21, MODS, METS, EAD тощо, які використовуються для опису різних типів бібліотечних ресурсів, а саме книжки, журнали, архіви, музика тощо.

4.4. Використання тегів для опису ресурсів. Поле тег у системі VuFind - це елемент форми, який дозволяє користувачу вводити чи вибирати теги для каталогізації чи пошуку ресурсів. Теги - це ключові слова чи фрази, які описують зміст чи характеристики ресурсу. Поле тег може бути різного типу. Наприклад, текстове поле, список, прапорець або перемикач. Тип поля тег визначається атрибутом типу тега <input>. Наприклад, <input type="text" name="tag"> створює текстове поле для введення поля тег.

Зареєстрований користувач може додавати свої власні теги в системі VuFind. Для цього потрібно перейти до будь-якого ресурсу, що його цікавить, і натиснути на кнопку *Додати тег* (рис. 6) Можна ввести свій тег у поле тег або вибрати із запропонованих варіантів тегів, що були внесені цим користувачем раніше.

[14] https://uk.wikipedia.org/wiki/LDAP

[15] Shibboleth Consortium https://www.shibboleth.net/

[16] http://dubincore.org

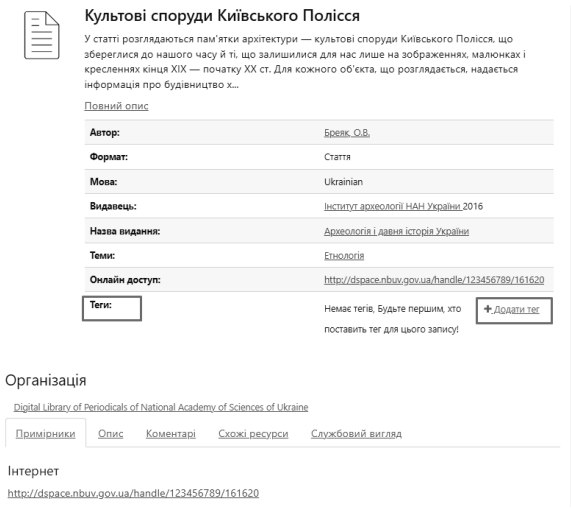

Рис. 6. Додати тег до ресурсу

Після цього тег буде збережений і відображений на сторінці ресурсу (рис. 7). Ви також можете переглянути всі свої теги в розділі *Ваш обліковий запис* і керувати ними. Функція додавання власних тегів дає змогу користувачу краще організувати свою колекцію, ділитися своєю думкою з колегами (іншими користувачами) та знаходити ресурси за темами, що його цікавлять.

Для того, щоб знайти ресурси за тегами в системі VuFind, можна використовувати такі способи:

– Скористатися хмарою тегів, яка відображається на сторінці *Збережені ресурси*. Хмара тегів показує найпопулярніші теги, присвоєні ресурсам користувачами. Можна вибрати будь-який тег у хмарі, щоб побачити список ресурсів із цим тегом[17].

– Ввести ім'я тега в поле пошуку на сторінці *Збережені ресурси* або на головній сторінці. Це дозволить знайти всі ресурси, які містять цей тег.

– Є можливість шукати за кількома тегами одночасно, використовуючи спеціальні символи. Якщо потрібно знайти ресурси, які мають усі зазначені теги, то використовують знак плюс (+) між іменами тегів. Наприклад, tags/tag1+tag2 знайде всі ресурси, які мають і tag1, і tag2. Якщо потрібно знайти ресурси, які мають хоча б один із зазначених тегів, то використовують кому (,) між іменами тегів. Наприклад, tags/tag1,tag2 знайде всі ресурси, які мають tag1 або tag2[18]

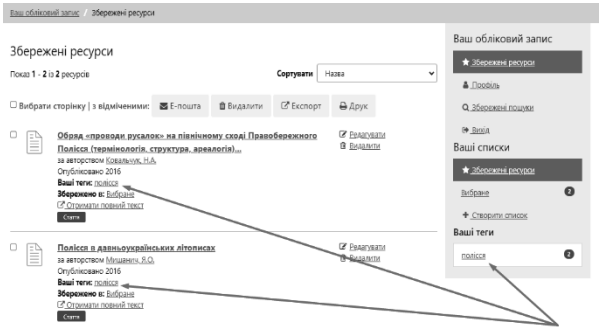

Рис. 7. Використання тегів для опису ресурсів

4.5. Пошук – основна функція VuFind для користувачів. Цей сервіс VuFind індексує всі наявні записи всіх бібліотек, підключених до системи, що дає змогу реалізувати простий і розширений пошук.

4.5.1. Простий пошук – це найбільш загальний і найзручніший тип пошуку, який дає можливість ввести один або кілька пошукових термінів у поле пошуку на головній сторінці й отримати список релевантних ресурсів, тобто ресурсів, які задовольняють ту чи іншу умову. У варіанті простого пошуку сформульований пошуковий вираз має відношення до відповідного набору описових полів (рис. 8), тобто на цей час у разі вибору опції *Всі поля* пошук здійснюється одночасно за:

– авторами, dc.creator ;
– назвами, dc.title;
– назвами журналу, dc.relation;
– описом або анотаціями статей, dc:description.abstract;
– предметами або тематичними розділами статей, dc.subject;
– тегами.

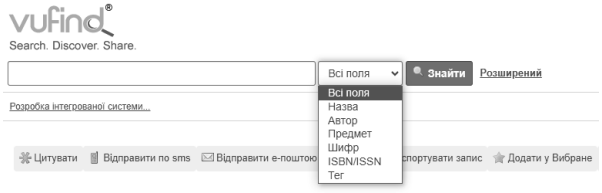

Рис. 8. Пошукові поля для простого пошуку

[17] https://www.ibm.com/docs/ru/ram/7.5.3?topic=assets-finding-using-tags

[18] https://ivanvetoshkin.me/2020/07/multiple-tags-search/

Звуження простору пошуку за тими чи іншими пошуковими полями здійснюється вибором відповідного поля зі списку *Всі поля*. У простому пошуку після введення щонайменше 3-х літер є пошукові підказки (рис. 9), обравши пошукову пропозицію, її можна застосувати до відповідного поля пошуку.

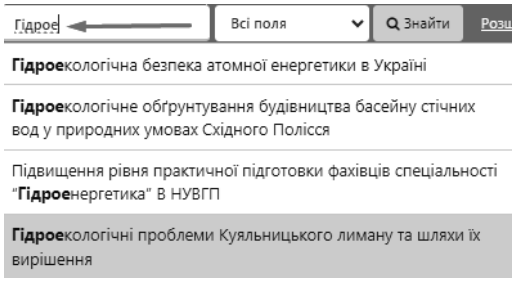

Рис. 9. Підказки для простого пошуку

Якщо ввести для пошуку слово "насіння", результуючий список документів за замовчуванням видається за релевантністю, тобто за ступенем відповідності запиту. Надається також можливість вибрати і відсортувати результат за іншим параметром, наприклад, за "роком видання", "автором" або "назвою". У запропонованому прикладі простого пошуку не всі знайдені документи мають у своїй назві слово "насіння", серед знайдених є й такі, де слово "насіння" зустрічається в анотації (рис. 10).

Тут ми розглянули лише приклад пошуку окремих слів. Інші питання, зокрема, пов'язані зі стоп-словами (слова, що не шукаються); пошук за багатьма словами; використання в словах групових символів; пошук за близькістю звучання слова; пошук за фразами; пошук з використанням відстані між словами; пошук за важливістю слів або фраз; обов'язкова наявність слів або фраз, тобто все те, що стосується простого пошуку і мови пошукових запитів розглянуто в *Інструкції користувача з перегляду та пошуку статей*, яка розміщена на порталі VuFind.

4.5.2. Розширений пошук – це більш точний і гнучкий тип пошуку, який дає змогу вказати різні критерії для пошуку ресурсів, а також обрати:

– тип пошуку (наприклад, усі слова, будь-яке слово або точний збіг);

– поля метаданих, такі як, заголовок, автор або тема та вказати їх значення;

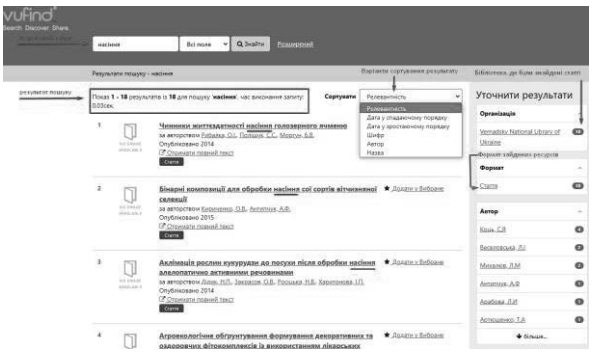

Рис. 10. Результати пошуку слова "насіння"

– логічний оператор (наприклад, І, АБО або НЕ);

– додавати додаткові рядки пошуку;

– також можна застосувати фасети для уточнення результатів за різними категоріями.

Розширений пошук має такі переваги:

– Більша точність. Розширений пошук дає змогу вказати точніші критерії для пошуку ресурсів, як-от тип пошуку, поле метаданих та логічний оператор. Це допомагає уникнути нерелевантних результатів і звузити межі пошуку.

– Більша гнучкість. Розширений пошук дозволяє комбінувати різні критерії для пошуку ресурсів, використовуючи кілька рядків пошуку і фасети. Це дає змогу створювати складні та різноманітні запити, які задовольняють різні потреби користувачів.

– Більша зручність. Розширений пошук надає зручний інтерфейс (рис. 11) для введення критеріїв пошуку, який містить списки, що випадають, чекбокси та радіокнопки. Це спрощує вибір потрібних параметрів і позбавляє від необхідності запам'ятовувати синтаксис запитів. У разі потреби можна знову повернутися до форми формування розширеного запиту і відредагувати його, або розпочати новий пошуковий запит. Результат пошуку (рис. 12) видається списком, де кожна його стаття представлена наступним чином:

– назва статті, яка є посиланням на окрему сторінку з описом цієї статті;

– список авторів із посиланнями на список робіт для кожного з них;

– рік публікації статті;

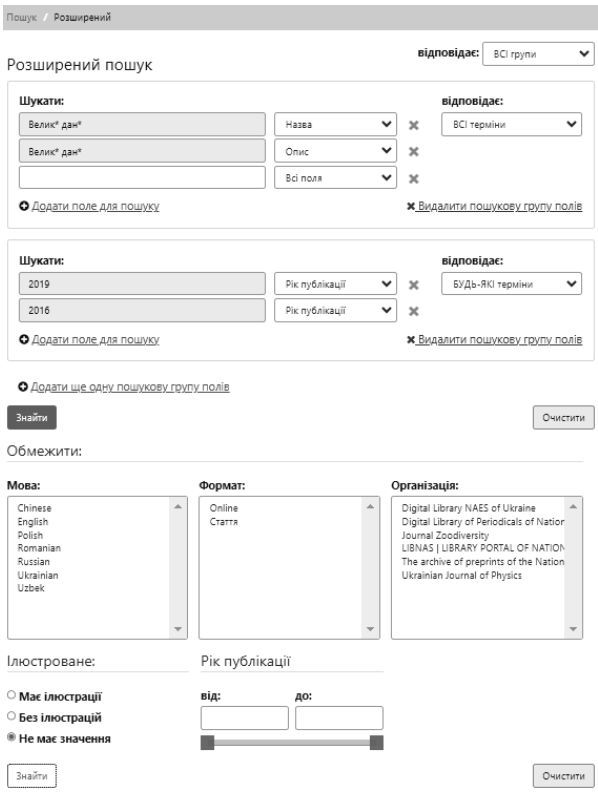

Рис. 11. Формування запиту в режимі розширеного пошуку

– посилання на джерело, звідки можна отримати повний її текст.

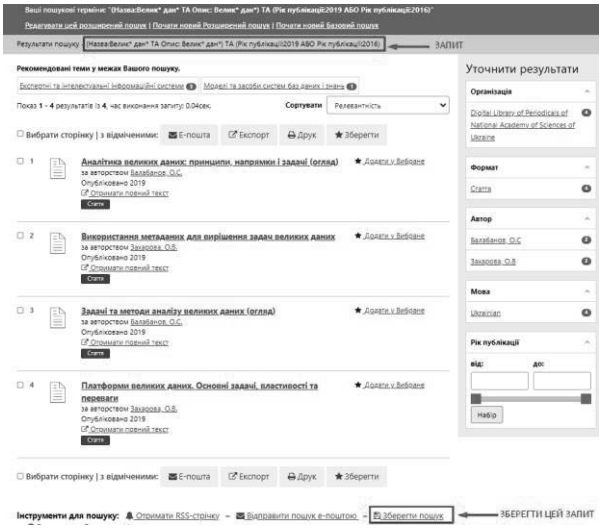

Рис. 12. Сторінка результату пошуку та опція Зберегти запит

Разом із результатом пошуку система видає перелік посилань на теми і кількість робіт на ці теми, що, можливо, пов'язані з результатом даного пошуку.

Для зареєстрованих користувачів є можливість зберегти кожну зі знайдених статей у власних колекціях документів (опція *Додати у вибране*).

Також праворуч списку результатів розташована панель *Уточнити результати*, де можна далі фільтрувати даний результат пошуку за різними чинниками (які називають фасетами), такими як організація, що надає свої ресурси в даний харвестр, автор, мова, тип ресурсу, рік публікації та ін.

Якщо у користувача є потреба відслідковувати результати свого запиту (добре налаштованого розширеного пошуку) й надалі або поділитися ним з колегами, цей запит можна зберегти.

Пошук з'явиться у списку *Збережених пошуків* для даного користувача (рис. 13). Крім власне пошукового запиту там зберігається час виконання цього пошуку, обмеження, які має цей пошук (наприклад, обмеження мови), кількість документів, знайдених у результаті його виконання, чи застосовувати розклад для розсилки даного пошуку, а також посилання на функцію його видалення.

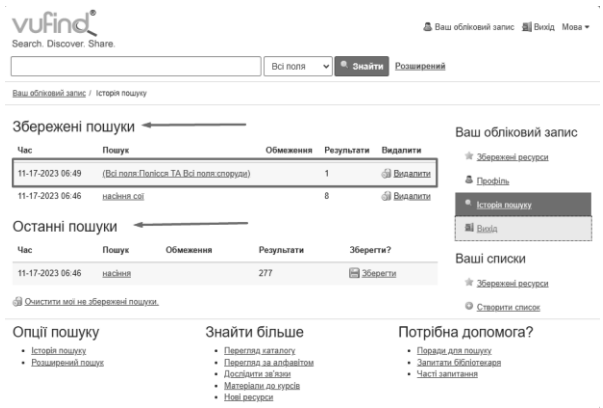

Рис. 13. Історія пошуку

Система відстежує всі пошукові запити, які виконувались цим користувачем і записує їх до списку, а також надає можливість їх збереження. Тобто, якщо користувач не зберіг якийсь важливий пошуковий запит, його можна зберегти і згодом знайти серед останніх пошуків.

4.6. Перегляд за фасетами в системі VuFind є потужним інструментом для уточнення і навігації за результатами пошуку. Фасети (або фасетні поля) являють собою категорії, які групують результати пошуку за певними атрибутами або метаданими. Наприклад, автор, організація, рік публікації, тема, тип ресурсу. Цей підхід

дає змогу користувачам точніше визначити параметри, що їх цікавлять, і звузити результати пошуку. Перегляд за фасетами в режимі реального часу оновлюється залежно від обраних користувачем параметрів, що дає змогу моментально бачити зміни в результатах пошуку. Можливість використовувати кілька фасетів одночасно допомагає користувачам проводити пошук, комбінуючи різні параметри для точніших і специфічніших запитів.

4.7. Авторитетний пошук широко застосовується в бібліотечних системах і наукових ресурсах, зокрема, і в системі VuFind, де стандартизація та ідентифікація авторів, термінів і тем є критично важливою для точного дослідження.

Ось декілька його аспектів:

Використання авторитетних файлів (Authority Files). Це списки, що містять стандартизовані ідентифікатори для імен авторів, термінів предметного покажчика та інших ідентифікаторів та контрольованих значень. Вони можуть містити імена авторів, варіанти їх написання, ідентифікаційні номери та інші метадані. Деякі бібліотечні системи (Koha, Alma, Sierra) дозволяють створювати і керувати своїми власними авторитетними файлами та застосовувати їх під час індексації та пошуку, інші інтегрують зовнішні авторитетні файли.

Нормалізація імен і термінів для дотримування стандартів і запобігання неоднозначності. Це містить й обробку варіантів написання, пов'язаних синонімів та інших форм для забезпечення однозначності ідентифікації одного й того ж автора або терміну в усіх результатах пошуку.

Уніфікація результатів запиту, тобто надання стандартизованих посилань на відповідні авторитетні файли. Це допомагає усунути дублікати, виявити та об'єднати різні записи про одного і того ж автора або термін та створити єдине подання інформації.

Використання унікальних ідентифікаторів, таких як ORCID для дослідників. Це допомагає пов'язувати ідентичних авторів та їхні роботи, навіть у разі різних варіантів написання імені.

Висновки

У цій статті ми спробували описати деякі можливості системи Vufind, виявлені і частково впроваджені в діючу систему під час виконання проєкту. Насамкінець слід наголосити про наступне. Система VuFind може відіграти важливу роль у покращенні доступу до академічних бібліотечних ресурсів:

– Вона може інтегрувати дані з великої кількості різних джерел, що розширює доступ до різноманітних матеріалів. Це включає в себе каталоги бібліотек, цифрові архіви, бази даних і онлайн-ресурси.

– Тут надаються ефективні інструменти для пошуку інформації. Користувачі можуть швидко знаходити необхідні ресурси, такі як книги, статті, журнали та інші матеріали, скорочуючи час пошуку.

– Користувачі можуть здійснювати пошук, об'єднуючи результати з різних джерел. Можливості фільтрації дають змогу уточнювати запити і знаходити потрібні ресурси з урахуванням різних параметрів.

– Система дозволяє інтегрувати авторитетні файли, забезпечуючи стандартизований ідентифікаційний пошук за авторами, термінами та іншими контрольованими значеннями.

І це далеко не всі можливості. Її гнучкість налаштувань, відкритий код і активне товариство дають нам упевненість у правильному виборі системи.

Література

1. Резніченко В.А, Новицький О.В., Проскудіна Г.Ю. Інтеграція наукових електронних бібліотек на основі протоколу OAI-PMH // Проблеми програмування. − 2007. − № 2 – С. 97–112. dspace.nbuv.gov.ua/hadle/123456789/291
2. Houser J. The VuFind implementation at Villanova Universityy // Library Hi Tech – 2009. − Vol. 27 No. 1, pp. 93-105. https://doi.org/10.1108/07378830910942955
3. Андон Ф.И., Матюхина К.Г., Новицкий А.В., Кудим К.А., Проскудина Г.Ю., Резниченко В.А. Функциональные возможности и статистика использования Научной электронной библиотеки периодических изданий НАН Украины // Проблеми програмування. − 2017. − № 3 – С. 68–95. https://pp.isofts.kiev.ua/index.php/ojs1/article/view/298

4. Новицкий А.В., Кудим К.А., Резниченко В.А., Проскудина Г.Ю. Создание научных архивов с помощью системы Eprints // Проблеми програмування. − 2007. − № 1 – С. 46–60. http://dspace.nbuv.gov.ua/handle/123456789/275
5. Кудім К.О., Резніченко В.А, Новицький О.В., Проскудіна Г.Ю., Овдій О.М. Розробка інтегрованої системи періодичних наукових видань на основі OJS// Проблеми програмування. − 2015. − № 3 – С. 72–85. https://pp.isofts.kiev.ua/ojs1/article/view/150
6. Резниченко В.А, Проскудина Г.Ю. О функции поиска в электронной библиотеке // Труды XII-й Всероссийской научной конференции «Электронные библиотеки: перспективные методы и технологии, электронные коллекции» RCDL2010 – Казань, Россия, 2010, С. 326-332. http://rcdl.ru/doc/2010/326-332.pdf
7. Dirk Pieper, Friedrich Summann Bielefeld Academic Search Engine (BASE) An end-user oriented institutional repository search service // Library Hi Tech – October 2006. DOI:10.1108/ 07378830610715473

References

1. Reznichenko V.A., Novitsky O.V., Proskudina G.Yu. Integration of scientific electronic libraries based on the OAI-PMH protocol // Problems in programming – 2007 – no. 2 – pp. 97–112. (In Ukraine). dspace.nbuv.gov.ua/hadle/123456789/291
2. Houser J. The VuFind implementation at Villanova Universityy // Library Hi Tech – 2009. − Vol. 27 No. 1, pp. 93-105. https://doi.org/10.1108/07378830910942955
3. Andon F.I., Kudim K.A., Matuhina K. G., Novitsky O.V., Proskudina G.Yu., Reznichenko V.A. Functionality and Statistics of the use of the Scientific digital library of periodical publications of the NAS of Ukraine // Problems in programming – 2017 – no. 3 – pp. 68–95. (in Russian). https://pp.isofts.kiev.ua/index.php/ojs1/article/view/298 (In Ukraine).
4. Novitsky O.V., Kudim K.A., Proskudina G.Yu., Reznichenko V.A. Create scientific archives using the Eprints system // Problems in programming – 2007 – no. 1 – pp. 46–60. (In Russian). dspace.nbuv.gov.ua/handle/123456789/275
5. Kudim K.O., Reznichenko V.A., Novitsky O.V., Proskudina G.Yu., Ovdii O.M. Development of integrated system of scientific periodicals based on the OJS // Problems in programming – 2015 – no. 3 – pp. 72–85. (In Ukraine). https://pp.isofts.kiev.ua/ojs1/article/view/150
6. Reznichenko V.A., Proskudina G.Yu. On the search function in an electronic library // Proceedings of the XII All-Russian Scientific Conference “Electronic Libraries: Advanced Methods and Technologies, Electronic Collections” RCDL2010 – Kazan, Russia, 2010, pp. 326-332. (In Russian). http://rcdl.ru/doc/2010/326-332.pdf
7. Dirk Pieper, Friedrich Summann Bielefeld Academic Search Engine (BASE) An end-user oriented institutional repository search service // Library Hi Tech – October 2006. DOI:10.1108/ 07378830610715473

Одержано: 19.11.2023

***Про авторів*:**

Проскудіна Галина Юріївна,
науковий співробітник.
Кількість публікацій в
українських виданнях – 33.
Кількість зарубіжних публікацій – 15.
http://orcid.org/0000-0001-9094-1565

Кудім Кузьма Олексійович,
молодший науковий співробітник.
Кількість публікацій в
українських виданнях – 20.
Кількість зарубіжних публікацій – 1.
http://orcid.org/0000-0001-9483-5495

Резніченко Валерий Анатолієвич,
кандидат фізико-математичних наук,
заступник завідувача відділом.
Кількість публікацій в
українських виданнях– 62.
Кількість зарубіжних публікацій – 4.
Індекс Хірша – 12.
http://orcid.org/0000-0002 4451-8931

Місце роботи авторів:

Інститут програмних систем НАНУ,
03187, Київ-187, пр. Академіка
Глушкова, 40.
E-mail: guproskudina@gmail.com;
kuzmaka@gmail.com; reznich@isofts.kiev.ua

УДК 004.94 http://doi.org/10.15407/pp2023.04.027

О. Захарова

МЕТАДАНІ НАУКОВИХ ДОКУМЕНТІВ ЯК СКЛАДОВА СИСТЕМИ ІНФОРМАЦІЙНИХ РЕСУРСІВ «ВІДКРИТОЇ НАУКИ»

Відкрита наука – це рух, мета якого полягає в отриманні більш доступних результатів досліджень, включаючи код, дані та наукові документи. Він охоплює багато різних, але часто пов'язаних аспектів, що впливають на весь життєвий цикл дослідження, включно із відкритим доступом до публікацій, дослідницькими даними, програмним забезпеченням з відкритим кодом і робочими процесами, суспільною наукою, із відкритими освітніми ресурсами та альтернативними методами оцінки досліджень. А також із відкритим рецензуванням, експертними оцінками тощо. Тобто, створюється наука, заснована на спільних принципах, які забезпечують порівняльність, сумісність і користь для всіх досліджень на планеті. Запорукою ефективного застосування та інтеграції ресурсів відкритої науки є їх структурований опис, що базується на принципах повноти та необхідності метаінформації, зручності її використання та інтероперабельності. Одиницями такого опису є метадані.

Дане дослідження не охоплює всього широкого спектру ресурсів відкритої науки. Його метою є структуризація та формалізація лише описів наукових документів як значущої частини наукових знань відкритої науки. Для реалізації поставленої цілі в статті визначено таксономію ресурсів відкритих наукових документів та запропоновано єдину інтегровану систему їхніх метаданих, яка містить як множину загальних характеристик, притаманних усім типам наукових документів, незалежно від їх призначення, форми чи формату представлення, так і ієрархічно поширювану систему специфічних ознак окремих типів ресурсів.

Ключові слова: ресурси відкритої науки, відкриті наукові знання, метадані наукових документів, схеми метаданих, метадані наукових публікацій, метадані освітніх матеріалів, таксономія понять відкритої науки, інфраструктура відкритої науки, тип відкритого ресурсу, метадані дисертацій, метадані монографій, формати представлення наукових ресурсів.

Вступ

Інституціоналізація відкритої науки зародилася кілька десятиріч тому як політика перетворення наукової практики з метою адаптації до змін, викликів, можливостей та ризиків цифрової ери та підсилення впливу науки на суспільство. Для забезпечення колективного розуміння його наслідків в усьому світі, Організація Об'єднаних Націй з питань освіти, науки та культури (ЮНЕСКО) в листопаді 2021 року ухвалила набір рекомендацій для міжнародної основи політики й практики *відкритої науки* із загальними цінностями, принципами та стандартами. ЮНЕСКО визначає відкриту науку як «інклюзивну конструкцію, яка поєднує різні рухи та практики, спрямовані на те, щоб зробити багатомовні наукові знання відкритими, доступними та придатними для повторного використання, розширити наукове співробітництво й обмін інформацією для блага науки й суспільства, а також відкрити процеси створення, оцінки й передачі наукових знань представникам суспільства за межами традиційної наукової спільноти».

Тобто відкрита наука – це прозорі та доступні знання, що розповсюджуються та розвиваються через спільні мережі. Але слід зазначити, що відкритий підхід не лише надає різні типи знань, що є у вільному доступі для всіх учасників спільноти для повторного використання, а й забезпечує цілісну інфраструктуру сумісних, стандартизованих послуг та апаратних засобів. Це сприяє дефрагментації, підтримці та спрощенню наукових практик і послуг, цілісності досліджень і наукової дипломатії в цілому, а також міжнародному науковому співробітництву. Створюється наука, заснована на спільних принципах, які забезпечують порівняльність, сумісність і користь для всіх досліджень на планеті. Сховища, ресурси, бази даних і сервіси проєкту стають актуальними для вчених у географічному масштабі за межами регіону, в якому вони розташовані, що приводить до розвитку потенціалу в глобальному масштабі.

Відкрита наука – це рух, метою якого є отримання більш доступних результатів досліджень, включно із кодом, даними та науковими паперами. Він охоплює багато різних, але часто пов'язаних аспектів,

що впливають на весь життєвий цикл дослідження, включно із відкритим доступом до публікацій, дослідницькими даними, програмним забезпеченням з відкритим кодом і робочими процесами, суспільною наукою, із відкритими освітніми ресурсами та альтернативними методами оцінки досліджень. А також із відкритим рецензуванням, експертними оцінками тощо.

Загальна таксономія понять Відкритої науки [1] наведена на рис.1.

Для забезпечення доступу до наукових знань вводиться поняття «інфраструктура відкритої науки». Мається на увазі віртуальна або фізична дослідницька інфраструктура загального користування включно із основним науковим обладнанням та набором інструментів, таких як колекції, журнали та платформи відкритого доступу до публікацій, сховища, архіви та наукові дані. А також сучасні інформаційні та відкриті наукометричні системи, що використовуються для оцінки та аналізу різних галузей науки. Крім того відкрита інфраструктура для обчислень та обробки даних дозволяє здійснювати колективний і мультидисциплінарний аналіз даних, вивчати об'єкти цифрової інфраструктури, які необхідні для підтримки відкритої науки і задоволення потреб різних товариств. Різні сховища даних мають бути адаптовані до конкретних характеристик об'єктів, розміщених в них (публікацій, даних або кодів), до місцевих умов, потреб користувачів та вимог дослідницьких спільнот, але водночас вони повинні застосовувати стандарти функціональної сумісності та передові методи роботи для забезпечення належної перевірки тих матеріалів, що розміщуються в сховищах та для полегшення їх пошуку й повторного використання. Але, одразу зазначимо, що відкрите апаратне забезпечення та інфраструктура, яка забезпечує доступ до відкритих знань, виходять за межі цього дослідження.

Рис.1. Таксономія Відкритої науки

Фокусом даного дослідження є лише структуризація та формалізація описів наукових документів як значущої частини наукових знань (ресурсів) відкритої науки за допомогою створення системи їхніх метаданих.

Таксономія ресурсів «відкриті знання»

Насамперед, слід визначитися, що ж саме ми будемо розуміти під науковими знаннями. Тобто, які типи ресурсів відкритої науки створюють «відкриті наукові знання».

Загалом, під науковим знанням розуміють набір надійної інформації про Світ, отриманої в процесі збору даних, експериментів та аналізу, досягнутих за допомогою наукових досліджень. На макро рівні всі типи наукових знань можна розділити на кілька великих груп: наукові документи, науковці (фізичні та юридичні особи), експонати (макети, моделі), наукові програми досліджень та наукові заходи. Аналіз джерел наукових знань дозволив детальніше класифікувати основні типи ресурсів відкритої науки у такий спосіб:

- Наукові документи
  - Наукові публікації
    - Монографії
    - Дисертації
    - Препринти
    - Автореферати
    - Наукові реферативні огляди
    - Тези
      - Бакалаврські
      - Магістерські
      - Кандидатські
      - Докторські
    - Інтернет-ресурси
    - Наукові звіти
    - Аналітичні огляди
    - Наукові видання (журнали, збірники тощо)
    - Статті
      - З журналів
      - Зі збірників
      - З газет
  - Матеріали форумів, конференцій, семінарів тощо
    - Наукові доповіді
    - Тези доповідей
    - Презентації
    - Матеріали стендів
  - Освітні матеріали
    - Матеріали курсів
    - Підручники
    - Лекції
    - Методичні та дидактичні матеріали
    - Презентації
    - Реферати
  - Патенти
  - Програмне забезпечення (ПЗ)
    - Проєкти, схеми, моделі даних, процесів тощо
    - Програмні застосунки та бібліотеки
    - Відкритий вихідний код ПЗ
    - Ліцензії
    - Набори даних
    - Івенти у програмних застосунках
  - Експертні оцінки
  - Рецензії
  - Свідоцтва про реєстрацію авторських прав
  - Інтерактивні ресурси
  - Мапи
  - Музичні нотатки
  - Аудіо матеріали
  - Зображення
    - Статичні рисунки
    - Відеоматеріали
  - Протоколи випробувань
  - Статистична інформація різних типів
    - Міжнародна музейна статистика
  - Закони, нормативні акти тощо
- Експонати, моделі, макети, історичні і культурні пам’ятки (цифрові описи з фотоматеріалами), матеріали музейних і бібліотечних архівів
- Експозиції, виставки
- Наукові програми

  - Освітні
  - Дослідницькі
  - Експериментальні
- Наукові заходи
  - Конференції
  - Симпозіуми
  - Семінари
  - Наради
  - Випробування
  - Виставки
- Наукові дослідники (фізичні та юридичні особи)
  - Науковці/дослідники (фізичні особи)
  - Наукові установи
    - Науково – дослідницькі інститути
    - Учбові центри
    - Конструкторські бюро
    - Університети
    - Музеї
    - Виставкові центри
    - Архіви
    - Наукові бібліотеки
    - Інші дослідницькі організації

Наукові документи є відкритими, до них забезпечується безкоштовний доступ, вони можуть багаторазово використовуватися, зберігатися та повторно поширюватись будь-якою особою за умови зазначення джерела. Для програмного забезпечення відкритий вихідний код в людино– і машино–читаному форматі надається користувачам у зручній для них формі відповідно до відкритої ліцензії. Це дає третім особам право доступу, зміни, доповнення, вивчення, використання та/або розповсюдження програмного забезпечення і його вихідного коду, дизайну або концепції, а також створення на їхній основі похідних робіт. Найважливіші аспекти відкритих даних пов’язані з їхнім змістом та внутрішньою організацією. В деяких ситуаціях також важливу роль відіграє третій аспект відкритих даних – їх оточення. Це найскладніший, але дуже важливий аспект для стратегічного й тематичного розвитку аналітики й пошуку знань, особливо з підключенням суміжної проблематики. В просторі відкритих даних контекстом для заданого набору є всі інші дані, з якими їх зможуть коректно зв’язати аналітики з тих чи інших підстав. Правильно визначити контекст можна лише, якщо для основних даних правильно задана предметна область (ПО) та їхнє призначення. Це досягається через опис даних набором спеціальних характеристик – метаданих.

Якість відкритих даних починається з якості їхніх метаданих. Зазвичай схеми метаданих визначаються, виходячи, перш за все, з таких аспектів даних, як сенс, структура та формат.

Метою даного дослідження є визначення схем метаданих для ресурсів «відкритих наукових документів» як складової інфраструктури відкритих ресурсів на основі існуючих схем та стандартів метаданих. Для кожної з виділених груп можна виокремити загальні метадані, які можуть бути використані для опису ресурсів усіх типів групи, та більш специфічні, що враховують особливисті окремих типів.

Різновиди і типи метаданих

Загалом, у поняття метаданих вкладається додатковий опис якогось – об'єкту або процесу. В ракурсі даних досліджень такими об’єктами/процесами є різноманітні компоненти відкритих наукових знань. Сам об'єкт метаданих може мати різні форми і класифікуватись за певними ознаками. Наприклад, зміст – опис шуканого об'єкта у вигляді розміру й типу файлу, інформації про вміст ресурсу, ставлення до ресурсу або його компонентів – базова інформація про об'єкт тощо.

Будь-яка інформаційна система зазвичай передбачає класифікацію метаданих на три великі головні групи: внутрішні – опис явних ознак самого об'єкту (розмір або тип файлу); адміністративні – інформація про об'єкт (автор, виконавець тощо); описові – інформація про природу об'єкту, його особливі ознаки, посилання на інші об'єкти, пов'язані з шуканим [2].

Формат метаданих являє собою уніфіковану форму опису властивостей об'єкта, що дозволяє отримати більш повне уявлення про цей об’єкт. На сьогодні розроблено велику кількіть стандартів мета-

даних для об'єктів різних типів, призначення та форматів представлення. Найпоширенішими є:

– MARC та його різновиди – стандарт, в основному застосовується для книг і бібліографічних ресурсів із зазначенням назви, автора, року написання або виходу.

– DCMI – стандарт, прийнятий для опису інтернет-об'єктів, електронних документів, ресурсів тощо.

– FOAF і vCard – опис персоніфікованих даних людей і організацій (у форматі vCard при експорті з мобільних пристроїв зберігаються списки контактів).

– CDWA – стандарт для опису історичних або музейних цінностей.

– ONIX і PRISM – інформація про видавництво.

– CIF – кристалографія;

– VICAR – обробка зображень, отриманих із супутників.

– NewsXML – новинні метадані тощо.

Цей список можна продовжувати до нескінченності, оскільки для будь-якого аспекту людської діяльності сьогодні можна знайти єдиний підхід в описі. Тому мета даного дослідження – на основі запропонованої вище (з можливим розширенням) класифікації ресурсів відкритої науки та на підставі аналізу існуючих схем метаданих, пошукових систем, стандартів опису, так чи інакше пов'язаних з тим чи іншим об'єктом наукових знань, сформувати єдині системи метаданих для кожної з груп наукових документів як ресурсів відкритої науки.

Базові аспекти побудови схеми метаданих наукових документів

Наукові публікації не лише складають більшу частину контенту відкритих документів просто за його обсягом, а й посідають центральне місце серед ресурсів даного типу. Адже зазвичай містять опис результатів наукових досліджень, а, отже, й нові наукові знання. Метадані наукових публікацій також є важливою частиною наукового дослідження, оскільки від них залежить показник цитування публікації. Українські та міжнародні журнали існують з відкритою, закритою та змішаною політикою доступу. В наукометричних базах даних, зокрема, Scopus та Web of Science, незалежно від рівня доступу, назва, анотація та ключові слова завжди відображаються вичерпно. Вони повинні виконувати функцію реклами, оскільки із загальної кількості публікацій, представлених у реферативних базах даних, дослідник обере саме інформативну. Більшість наукових публікацій являють собою різновиди бібліографічних ресурсів, для опису яких найпоширенішими стандартами, на сьогодні, лишається MARC [14] та його різновиди. Також, як і для решти типів контентів та інтернет-ресурсів, значущими лишаються загальні характеристики, визначені DCMI (Dublin Core Metadata Initiative) стандартом.

Але, зрозуміло, що цілісність наукових знань не можлива без усього широкого спектру наукових документів. Так аналітичні звіти та огляди не є опублікованими матеріалами, але зазвичай містять більш ємні та детальні описи результатів, ніж опубліковані за темою статті, препринти, монографії тощо. Дисертаційні роботи також мають особливості, як підсумок дослідження, проведеного з метою здобуття наукового ступеню автором. Зокрема, їх опис має враховувати вимоги та стандартні протоколи, застосовані до робіт даного типу. Недарма, в цифрових наукових бібліотеках дисертаційні роботи та автореферати до них виділяються в окремі репозиторії та пов'язані з певними критеріями пошуку у сховищі.

На відміну від дисертації або наукового звіту, які є суто науковою публікацією, монографії [21] належать до науково-популярної або навчально-методичної літератури. Розділи монографії можуть містити наукові положення, як-от, дисертації, але, водночас, монографія адресована ширшій аудиторії, а не лише фахівцям з теми. Тож має бути адаптована для читача і, за можливості, викладена простішою та зрозумілішою мовою.

Препринти щодо інших наукових, науково-популярних та навчально– методичних публікацій (статей, тез, монографій тощо), фактично є попереднім варіантом наукового видання, яке публікують ще до

випуску офіційного кінцевого варіанту матеріалу. Це роботи без рецензії та публікації в журналі. Їх ніяк не враховують у звітах, статистиці, вони не впливають на науковометричні показники вченого. Електронні препринти розміщені в Інтернеті, називаються "Е-принтом". Із 2017 року препринти введені до профілю Scopus [24]. На відміну від книжок (монографій) для препринту обов'язковими характеристиками є код та установа, де відбуваються відповідні дослідження.

Однак більшість наукових публікацій все ж вимагають попереднього рецензування науковими експертами. Такі рецензії виявляють актуальність досліджень, напрямки розвитку й зауваження до них, і самі є науково змістовними та у поєднанні з об'єктом рецензування значущими елеменами наукових знань. Їхні описові характеристики мають формуватися, виходячи з вимог щодо наукового рецензування, та враховувати загальноприйняту структуру побудови наукової рецензії.

Особливої уваги заслуговують також архівні документи, що складають чималу частину наукових знань та є підгрунтям сучасних знань. Основний міжнародний стандарт архівного опису – ISAD(G) [12]. Даний стандарт вводить правила багаторівневого опису архівних документів, визначає базову термінологію опису та пропонує систему характеристик, організованих у сім груп описової інформації (одиницею опису в даному випадку є документ будь-якого типу чи група документів, що розглядається як єдине ціле). Слід зазначити, що стандарт спрямований саме на опис архівних документів, тому містить також характеристики, які визначають фонди, архівну інформацію тощо.

Окрему групу наукових публікацій складають освітні матеріали, до яких належать: курси лекцій, матеріали навчальних курсів, підручники, е-підручники, методичні та дидактичні матеріали, презентації, реферати тощо. Частина з перелічених типів ресурсів, таких, як підручники, методичні та дидактичні матеріали тощо, передбачають наявність паперового варіанту. Решта зазвичай існує лише в цифровому представленні. Для освітніх матеріалів, що мають паперове та цифрове представлення, застосовуватиметься більшість метаданих, що є дійсними для монографій та статей. А для існуючих лише у цифровому представлені, – ті, що визначаються для інтернет-ресурсів, зокрема, е-підручників [19, 20].

Окрім специфіки, пов'язаної з типом відкритого ресурсу, певні особливості можуть бути обумовлені форматом його представлення. Головні три форми представлення наукових документів – це текст, зображення та презентації. Так, зокрема, діаграмам, схемам, фотоматеріалам, рисункам, що можуть бути як окремими повноцінними об'єктами наукових документів, так і приєднаним вкладенням або частиною більш складного об'єкта (наукової роботи), притаманні метадані зображень.

Усі ці особливості обумовлюють визначення специфічних метаданих для різних типів ресурсів у межах групи «науковий документ», разом із загальними описовими характеристиками, притаманними всьому спектру відкритих ресурсів цієї групи.

Отож, система метаданих «відкритих наукових документів» буде базуватися на:

– стандартах загального опису документів, головним з яких досі є DCMI (Dublin Core Metadata Initiative) [3]

– аналізі систем описових характеристик існуючих цифрових бібліотек, зокрема, описів дисертаційних робіт та авторефератів, електронних підручників, монографій тощо, та наборах критеріїв пошуку в них [15-17, 20]

– стандартах із бібліотечної справи та суміжних областей [23],

– загальному аналітичному аналізі існуючих міжнародних дескриптивних стандартів [13],

– аналізі систем індексації публікацій науковометричними базами даних Scopus [18], Web of Science тощо,

– міжнародному стандарті архівного опису – ISAD(G),

– стандартах і схемах метаданих зображень [4], таких як EXIF [5], DCF [6], IPTC, XMP [7 – 9], ISO 16684-2:2014,

– вивченні особливостей системи індексації монографій [22].

Необхідно усвідомлювати, що перелічені типи ресурсів тісно пов'язані з

іншими різновидами відкритих ресурсів, які також можна розглядати як різновиди відкритих наукових знань та з їх описами. Такими ресурсами, наприклад, є наукові видання, фізичні та юридичні особи – науковці тощо. Із побудовою єдиної системи метаданих відкритої науки такі взаємозв'язки дозволять повніше та багатогранніше здійснити опис відкритого наукового ресурсу. Так, наприклад, посилання на автора в метаописі наукової публікації дозволить отримати доступ до вичерпного метаопису даного науковця. Розгорнута система характеристик видання, де опубліковано наукову роботу, також може свідчити щонайменше про її важливість. Це павутиння розширюється зв'язками з відкритими ресурсами інших типів, як-от, наукові заходи, апаратне обладнання тощо.

Метадані відкритих наукових документів

Базуючись на вище викладених аспектах, існуючих стандартах та цифрових системах роботи з інформацією (пошукових системах, цифрових бібліотеках тощо), пропонується схема метаданих наукових документів, класифікованих за спільними та особливими ознаками типів ресурсів, поєднаних поняттям «відкритий науковий документ». Загальні метадані притаманні всім типам інформаційних ресурсів, що розглядаються.

Загальні метадані відкритих наукових документів:

- ідентифікатор інформаційного ресурсу (посилання на ресурс);
- назва ресурсу;
- тип ресурсу (наукова публікація/освітні матеріали/діаграма/модель тощо);
- дата останнього оновлення метаданих інформаційного ресурсу;
- дата реєстрації інформаційного ресурсу;
- служба, що реєструє інформаційний ресурс;
- інформація про власника ресурсу;
  - ПІБ рідною мовою;
  - ім'я та прізвище англійською мовою;
- консультант – особа, до якої можна звертатися за додатковою інформацією;
- інформація про авторів (може співпадати з власником)/співавторів
  - ПІБ рідною мовою;
  - ім'я та прізвище англійською мовою;
  - посада;
  - звання;
  - наукова ступінь;
  - посада, звання, науковий ступінь авторів англійською мовою (переклад згідно з міжнародними стандартами);
  - міжнародні ідентифікатори (коди) вчених:
    - ORCID ID;
    - Scopus ID;
    - Researcher ID;
    - h-індекс;
- додаткові відомості про авторів/співавторів:
  - місце роботи:
    - назва установи;
    - адреса;
  - контактні телефони;
  - електронна адреса;
  - кількість публікацій:
    - в українських виданнях;
    - в зарубіжних виданнях;
- коди рубрикатора – тематика ресурсу, що виражена кодами стандартного переліку тематичних рубрик;
- ключові слова;
- стислий опис (зміст ресурсу, влючно із анотацією чи рефератом, або опис контенту візуальних, аудіо– або мультимедійних ресурсів);
- фінансування – форма фінансування під час створення та введення ресурсу;
- дата створення/подання або останнього оновлення контенту;
- період оновлення;
- чи існує паперовий варіант;
- інформація про інтернет –ресурс, де опубліковано (цифрова бібліотека/електронний журнал тощо);
- інформація про файл ресурсу:
  - розмір;
  - формат;
  - місце зберігання у файловому сховищі;
- посилання на ресурс в мережі;

- метричні показники документу:
  - розмір сторінки;
  - шаблон;
  - кількість сторінок у документі.

Метадані текстових ресурсів:

- мова (для текстового ресурсу);
- УДК (код класифікатора);
- анотація мовою оригіналу;
- анотація англійською мовою;
- ключові слова рідною та англійською мовами;
- список використаних джерел (references):
  - посилання;
  - статус (закрите, відкрите, обмежене);
- відсоток відкритих посилань (open references), тобто доступних усім користувачам усіх API та сервісів Crossref;
- Funder Registry ID (назва та ідентифікатор організації, яка фінансує дослідження та є у спеціальному реєстрі Crossref);
- Funding Award Numbers (інформація щодо фінансування дослідження, наприклад, номеру грантів або премій, пов'язаних з науковим дослідженням);
- чи застосовується сервіс, що відслідковує зміни статті після публікації;
- повнотекстове посилання для сервісу Crossref Similarity Check (перевірка подібності, перевірка на плагіат);
- повнотекстове посилання для антивірусної бази iThenticate;
- посилання у списках літератури (Citation);
- додаткові метричні показники:
  - номери сторінок у журналі/збірнику;
  - кількість використаних джерел;
  - обсяг приєднаного матеріалу:
    - графічного (кількість рисунків);
    - таблиць;
    - формул;
    - схем;
- структура матеріалу (зміст):
  - глави/розділи/лекції;
  - додатки.

Додаткові метадані, специфічні для тез/дисертацій/авторефератів/дипломних робіт:

- на здобуття якого ступеня (кандидатський, докторський, магістерський тощо);
- код спеціальності ВАК;
- керівник досліджень;
- перелік друкованих робіт за темою досліджень;
- інформація про джерела фінансування дослідження (грант, стипендія тощо);
- дата захисту (для дисертації);
- місце захисту (для дисертації);
- дата присвоєння наукового ступеня (для дисертації);
- інформація про практичне впровадження;
- список використаних джерел англійською мовою (references);
- політика доступу;
- посилання на автореферат (дисертації).

Метадані, специфічні для звітів про науково-дослідну роботу:

- тип звіту (анотований, остаточний, річний тощо);
- період виконання досліджень;
- назва проєкту, в межах якого виконуються дослідження;
- кількість наукових робіт, опублікованих у рамках звітного періоду.

Додаткові метадані для наукових робіт, що мають паперовий варіант:

- видавництво, що опубліковало статтю (не визначається для інтернет-ресурсів);
- інформація про видання/журнал/газету (де опубліковано);
- дата публікації.

Специфічні метадані матеріалів конференцій/семенарів/форумів:

- тип події (конференція, семінар, форум тощо);
- назва події;
- назва матеріалів (збірника матеріалів);
- регулярність проведення;
- дата проведення події;
- місце проведення події:
  - країна;
  - місто;
  - установа;
  - адреса;
- загальна наукова спрямованість;
- інформація про наукові розділи:

- назва розділу;
- кількість доповідей в розділі;
- інформація про доповіді:
  - назва;
  - інформація про доповідача;
  - відомості про співавторів;
  - тип доповіді (стендова, повідомлення, повноцінна тощо);
  - анотація;
  - тези доповіді;
  - посилання на повний текст доповіді;
  - посидання на аудіозапис/відеозапис доповіді;
  - приєднані матеріали:
    - наявність;
    - тип (презентації, додатки, роздаткові матеріали, стендові матеріали тощо);
    - назва;
    - посилання на приєднані матеріали.

Специфічні метадані для публікацій (монографій/підручників/препринтів):

- рецензенти:
  - посада;
  - вчене звання;
  - науковий ступінь;
  - місце роботи;
  - ПІБ;
- рекомендація до видання;
- знак охорони авторського права;
- унікальний ідентифікатор DOI (роботи в цілому або розділів монографії).

Метадані, специфічні лише для препринтів:

- код препринту;
- установа.

Додаткові метадані, притаманні для освітніх матеріалів:

- тип матеріалу (підручник, методичні вказівки тощо);
- тип представлення (текст/ презентація/відео тощо);
- міністерство (чи інша керуюча установа);
- назва навчального закладу;
- до якого освітнього курсу належить;
- вихідні відомості.

Метадані, специфічні для рецензій:

- назва рецензованої роботи;
- посилання на науковий матеріал, що рецензується;
- дата рецензії;
- дані про автора рецензії:
  - посада;
  - вчене звання;
  - науковий ступінь;
  - місце роботи;
  - ПІБ;
- висновок (позитивний/негативний);
- посилання на повний текст рецензії;
- стислий зміст:
  - переваги рецензованої роботи;
  - зауваження до роботи.

Метадані, специфічні для архівних документів:

- характеристики ідентифікації:
  - рівень структури документа (наприклад, збірка робіт, матеріали конференції чи стаття, опис колекції чи конкретного експонату);
  - носій, де зберігається документ;
- характеристики контексту:
  - біографічні дані автора;
  - архівна історія документу (для архівних документів);
  - безпосереднє джерело комплектування або переклад (джерело, звідки було отримано документ, або дата та/або спосіб комплектування, якщо ця інформація частково або повністю конфеденційна);
- характеристики змісту та структури:
  - рамки (період часу, географія тощо) та зміст (форма документа, тема тощо) документа відповідно до рівня опису;
  - інформація щодо оцінювання, знищення документу або планування відповідних дій;
  - подальші поповнення (доповнення) одиниці опису;
- характеристики доступу та використання:
  - правила доступу (правовий статус або інші правила, що регламентують доступ до документу);
  - умови відтворення документу;
  - фізичні характеристики та технічні умови (апаратні та програми, необхідні для доступу до документу, вимоги до зберігання тощо);

  - науково-довідковий апарат (посилання на наявні довідкові засоби, які містять інформацію із контексту та змісту документу);
- характеристики взаємопов'язаних матеріалів:
  - наявність та місце розміщення оригіналу документу;
  - наявність та місцезнаходження копій;
  - пов'язані документи;
  - примітки про наявні публікації (публікації, що присвячені даній одиниці опису або засновані на її використанні);
- примітки (інформація про документ, що не може бути подана в інших характеристиках);
- характеристики контролю опису:
  - примітки архівіста (як і ким було створено опис документу);
  - правила, за якими створений опис;
  - дати створення опису.

Метадані зображень:

- розмір зображення в пікселях;
- тип зображення (цифрова фотографія, рисунок, схема, діаграма, мапа);
- опис зображення;
- коментарі користувача;
- права доступу до інформаційного об'єкту;
- приєднане аудіо (якщо є, ідентифікатор, назва та посилання);
- приєднана анотація (якщо є, ідентифікатор, назва та посилання);
- кольорове/чорно-біле;
- характеристики, притаманні лише цифровим фотографіям [10]:
  - параметри фотокамери, з якої було зроблено фотографію (з повним переліком можна ознайомитися в описі EXIF-схеми):
    - модель;
    - виробник;
    - діафрагма;
    - витримка;
    - швидкість;
    - експокорекція;
    - фокусна відстань;
    - світлосила;
    - експозамір;
    - відстань до об'єкта;
    - спалах режим;
    - спалах енергія;
    - тощо;
  - геолокація, де була зроблена фотографія;
- семантичні текстові анотації:
  - текстові мітки, приєднані до зображення;
  - назва рисунку, що виводиться на самому зображенні чи приєднана до нього;
  - розширений опис;
  - опис основних елементів зображення;
- характеристики візуалізації:
  - набір візуальних слів [11].

Метадані, що специфічні для презентацій:

- кількість слайдів;
- шаблон презентації, що використовується;
- розширений опис;
- пов'язаний файл доповіді;
- подія, до якої стосується презентація;
- спецефекти, що використовуються для слайд-шоу в цілому;
- метадані слайдів:
  - номер слайду;
  - тема слайду;
  - текстові мітки, приєднані до слайду;
  - рисунки/діаграми/таблиці, вставлені в слайд;
  - використані спецефекти.

Висновки

Мета дослідження полягає у розробці єдиної схеми метаданих ресурсів відкритої науки. Зокрема, фокусом даної роботи є визначення системи характеристик, які описують загальні та специфічні ознаки різних типів наукових документів як значущої частини наукових знань відкритої науки.

В результаті досліджень:

1) здійснено аналіз головних понять, визначень та потреб відкритої науки;
2) на базі цього аналізу виокремлені ресурси відкритої науки, що фор-

мують клас «відкриті наукові знання», та запропоновано таксономію цих ресурсів;

3) здійснено аналіз ресурсів у межах визначеної ієрархії «відкриті наукові знання» та базі цього аналізу запропоновано систему схем метаданих таких ресурсів.

Напрямки подальших досліджень полягають у вдосконаленні запропонованих схем метаданих і проведенні всебічного аналізу програмних засобів створення, управління та інтеграції метаданих для ресурсів відкритої науки та визначення можливості й ефективності їх застосування для тих чи інших груп ресурсів.

Література

1. https://www.fosteropenscience.eu/content/open-science-training-handbook
2. https://hi-news.pp.ua/tehnka-tehnologyi/7373-metadan-ce-viznachennya-vidi-ta-varanti-vikoristannya-metadanih-u-prikladnomu-programuvann.html
3. https://www.dublincore.org/
4. https://habr.com/ru/post/93119/
5. https://www.exif.org/category/specifications
6. http://exif.org/dcf.PDF
7. https://helpx.adobe.com/after-effects/using/xmp-metadata.html
8. ISO 16684-1:2012, Graphic technology – Extensible metadata platform (XMP) specification – Part 1: Data model, serialization and core properties
9. https://www.adobe.com/devnet/xmp.html
10. https://www.ixbt.com/digimage/metadxph.shtml
11. J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos. In Proc. of 9th IEEE Int'l Conf. on Computer Vision, Vol. 2, 2003.
12. 1999_ISAD_G.pdf (archives.gov.ua)
13. https://undiasd.archives.gov.ua/doc/ao-deskreptyvni%20standarty.pdf
14. https://www.loc.gov/marc/
15. https://nim.media/articles/shcho-take-metadani-statti-yak-voni-vplivayut-na-yiyi-indeksatsiyu
16. http://openscience.in.ua/key-metadata-elements.html
17. http://www.4author.com/ru/blog/story/metadannye-nauchnoi-stati-osnova-raboty-kotoruiu-budut-tcitirovat
18. http://www.library.univ.kiev.ua/ukr/res/scopus_author.pdf
19. https://dspace.uzhnu.edu.ua/jspui/handle/lib/38156
20. Каталоги – НБУВ Національна бібліотека України імені В. І. Вернадського (irbis-nbuv.gov.ua)
21. https://ukrlogos.in.ua/ua_monographie.php
22. https://isg-konf.com/uk/indeksatsiya-monografij-ukr/
23. http://www.nbuv.gov.ua/sites/default/files/basicpage_files/202107_basicpage_files_mat/perelik_standartiv-_2021.07.01.pdf
24. https://nim.media/articles/nova-funktsiya-v-scopus---preprinti

References

1. https://www.fosteropenscience.eu/content/open-science-training-handbook
2. https://hi-news.pp.ua/tehnka-tehnologyi/7373-metadan-ce-viznachennya-vidi-ta-varanti-vikoristannya-metadanih-u-prikladnomu-programuvann.html
3. https://www.dublincore.org/
4. https://habr.com/ru/post/93119/
5. https://www.exif.org/category/specifications
6. http://exif.org/dcf.PDF
7. https://helpx.adobe.com/after-effects/using/xmp-metadata.html
8. ISO 16684-1:2012, Graphic technology – Extensible metadata platform (XMP) specification – Part 1: Data model, serialization and core properties
9. https://www.adobe.com/devnet/xmp.html
10. https://www.ixbt.com/digimage/metadxph.shtml
11. J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos. In Proc. of 9th IEEE Int'l Conf. on Computer Vision, Vol. 2, 2003.
12. 1999_ISAD_G.pdf (archives.gov.ua)
13. https://undiasd.archives.gov.ua/doc/ao-deskreptyvni%20standarty.pdf
14. https://www.loc.gov/marc/
15. https://nim.media/articles/shcho-take-metadani-statti-yak-voni-vplivayut-na-yiyi-indeksatsiyu
16. http://openscience.in.ua/key-metadata-elements.html
17. http://www.4author.com/ru/blog/story/metadannye-nauchnoi-stati-osnova-raboty-kotoruiu-budut-tcitirovat
18. http://www.library.univ.kiev.ua/ukr/res/scopus_author.pdf
19. https://dspace.uzhnu.edu.ua/jspui/handle/lib/38156

20. Catalogs – NBUV National Library of Ukraine named after V. I. Vernadskyi (irbis-nbuv.gov.ua)
21. https://ukrlogos.in.ua/ua_monographie.php
22. https://isg-konf.com/uk/indeksatsiya-monografij-ukr/
23. http://www.nbuv.gov.ua/sites/default/files/basicpage_files/202107_basicpage_files_mat/perelik_standartiv-_2021.07.01.pdf
24. https://nim.media/articles/nova-funktsiya-v-scopus---preprinti

Одержано: 24.10.2023

Про автора:

Захарова Ольга Вікторівна,
кандидат технічних наук,
старший науковий співробітник.
Кількість наукових публікацій
в українських виданнях – 35.
http://orcid.org/0000-0002-9579-2973.

Місце роботи автора:

Інститут програмних систем
НАН України,
проспект Академіка Глушкова, 40.
Тел.: 526 5139.
E-mail: ozakharova68@gmail.com.
Моб.тел.: +38(068)594756

УДК 681.3 http://doi.org/10.15407/pp2023.04.039

Ю.В. Рогушина

ТРИВИМІРНА МОДЕЛЬ СЕМАНТИЧНОГО ПОШУКУ: ЗАПИТИ, РЕСУРСИ ТА РЕЗУЛЬТАТИ

Запропонована тривимірна модель семантичного пошуку , що аналізує пошукові запити, інформаційні ресурси та результати пошуку, пропонується як додатковий інструмент опису та співставлення інформаційно-пошукових систем (ІПС), що використовують різноманітні елементи штучного інтелекту та менеджменту знань для більш ефективного та пертинентного задоволення інформаційних потреб користувачів. Потрібно відмітити, що значення параметрів, які аналізує ця модель, не є взаємовиключними, тобто та сама ІПС може підтримувати кілька варіантів пошуку. Крім того, засоби подання запитів та ресурсів не завжди є порівнюваними. Проаналізовано існуючі підходи до семантизації пошукових запитів та використання зовнішніх джерел знань для їх виконання.

Модель дозволяє виявляти ІПС, для яких перетинаються тріади "запит-ІР-результат", та виконувати їх порівняння саме на цих підкласах пошукових задач. Це дозволяє визначати, які алгоритми пошуку виявляються більш пертинентними для конкретних задач користувачів і на основі цього обирати такі сервіси як джерело інформації для подальшої обробки. Важливою особливістю запропонованої моделі є використання лише тих характеристик ІПС, які можуть бути проаналізовані користувачами.

Ключові слова: семантичний пошук, онтологія, пошуковий запит.

Вступ

Пошук інформації (у локальних і глобальних мережах, на окремому комп'ютері) сьогодні є одним із найпоширеніших завдань, що входить до складу різних застосовних систем. Швидке зростання обсягу інформації, яку потрібно обробляти, та ускладнення її структури зумовлюють усе більшу потребу в розвитку засобів знаходження відомостей, необхідних користувачам для виконання їхніх завдань.

Семантичний пошук (СП) – термін, який використовують для позначення набору методів, призначених для покращення пошуку в документіах або у базі знань. На відміну від традиційних методів пошуку, зосереджених на ранжуванні документів на основі набору ключових слів (як у запиті користувача, так і в індексованому контенті), методи СП спрямовані на те, щоб враховувати контекст і семантику як запиту користувача, так і тих ресурсів, в яких здійснюється пошук, за допомогою використання засобів обробки природної мови, технологій Semantic Web та методів машинного навчання для отримання більш релевантних результатів.

Система семантичного пошуку (ССП) – це інформаційна система, що забезпечує пошук і розпізнавання інформаційних об'єктів (ІО) різних типів із використанням знань для зіставлення запиту з наявними інформаційними ресурсами на семантичному рівні [1]. ССП можна розглядати як певну інтелектуальну надбудову над традиційними *інформаційно-пошуковими системами* (ІПС) як загального призначення, так і спеціалізованими.

Актуальність проблеми семантизації пошуку

Багато дослідників звертають увагу на різні аспекти розвитку СП та критерії їх оцінювання [2] . Деякі з них докладно аналізують окремі аспекти СП: наприклад, зосереджуються на ПМ-запитах до баз знань або онтологій RDF [3].

Класичні інформаційно-пошукові системи (ІПС) отримують вхідні дані у вигляді запиту користувача, що подається як список ключових слів, а як вихідні дані генерують впорядкований список документів, релевантних цим ключовим словам.

Пошук у природномовних ресурсах

Методи обробки ПМ застосовуються для розуміння семантики запиту або документів, для розпізнавання частини мови (Part-Of-Speech – POS) у текстовому контенті (наприклад, визначити граматичні теги,

такі як іменник, сполучник або дієслово до окремих слів). Такий аналіз дає точні результати для повних, правильно сформованих речень [4], але набагато складніший для коротких текстів, таких як запити [5]. Теги POS можна використовувати також для таких задач, як: 1) розрізнення текстових ключових слів, наприклад, для розпізнавання іменованих об'єктів (Named-Entity Recognition – NER), де завдання полягає в тому, щоб визначити, які слова відповідають екземплярам об'єктів реального світу; 2) розділення співпосилання (co-reference resolution), тобто виявлення всіх ключових слова, що посилаються на ту саму сутність у тексті. Синтаксичний аналіз речень (parsing) виводить аналіз ПМ на наступний рівень, охоплюючи загальну структуру речень (як правило, через дерево аналізу залежностей).

Методи обробки ПМ часто поєднуються з лексичними базами знань для того, щоб ідентифікувати об'єкти в текстовому запиті або контенті та зіставити їх із відповідними об'єктами у базі знань (або іншому зовнішньому джерелі інформації про предметну область, такі як вікіресурси) для покращення результатів пошуку. Наприклад, WordNet використовується [6] для усунення неоднозначності слів ПМ. Аналіз структури Вікіпедії дозволяє краще оцінювати відповідність об'єктів пошуковому запиту [7] або для ідентифікації сутностей під час пошуку у колекціях документів [8] , тоді як [9] обробляють згадки сутності у Вікіпедії разом з іншими характеристиками документів.

Концептуально подібні підходи використовуються й у контексті Semantic Web – замість вікіресурсу аналізуються більш структуровані інформаційні джерела, такі як онтології, для отримання інформації щодо структури та екземплярів інформаційних об'єктів для кращого розуміння контексту запитів або сутностей, описаних в ПМ-документах. Наприклад, [10] пропонують використовувати онтології для інтерпретації ПМ-запитів, перетворюючи на основі дескриптивної логіки набір ключових слів на кон'юнкцію понять. Інші дослідника [11] використовують об'єкти та семантичні зв'язки з бази знань DBpedia для групування результатів пошукової системи у більш значущі групи.

Методи машинного навчання часто використовуються у СП для визначення семантики слів або сутностей у документах на основі гіпотези семантичної подібності слів, які зустрічаються в подібних контекстах. Ранні підходи в цій царині пов'язані з побудовою багатовимірних матриць, де кожне слово представлене розрідженим вектором у просторі великої розмірності. Зараз використовуються методи аналізу близькості слів на основі щільних векторних [12], які визначають максимальну ймовірність спільної появи слів у певному контексті за допомогою нейронної мережі. Це дозволяє генерувати векторні представлення слів із великих текстових корпусів для машинного навчання.

Пошук у базах знань

Деякі підходи до СП спрямовані на обробку декларативних баз знань (таких як онтології або графи знань), а не на колекцій ПМ-документів. Такі бази знань можуть бути подані багатьма різними способами, але наразі більшість існуючих реалізацій базується на стандартах Semantic Web – RDF та OWL. Відомі приклади таких баз знань – Google Knowledge Graph [13] , Dbpedia [14] та Wikidata [15] . Користувачі можуть будувати запити до таких джерел інформації як до традиційних ІПС (наприклад, як набір ключових слів) або структуровано (наприклад, SPARQL-запити) .

У багатьох дослідженнях, пов'язаних із проєктом Semantic Web [16, 17], аналізується формальний підхід до виконання структурованих запитів до онтологій, де дескриптивна логіка використовується для ранжирування результатів пошуку. Для покращення пошукових запитів, орієнтованих на онтології, можуть застосовуватися мета-онтології: як-от, у [18] WordNet використовується для зіставлення елементів онтології з лексичними об'єктами.

У роботах [19] та [20] розглянуто пошук у ресурсах спеціалізованих інформаційних об'єктів, які характеризуються

типами або відношеннями на основі природномовних запитів або ключових слів.

Гібридні підходи до пошуку використовують як текстовий, так і структурований контент: ІПС доповнює запити з ключових слів шляхом дослідження онтологічного графу [21], пошук використовує текстові метадані про об'єкти в структурованому сховищі [22].

Пошук на основі векторного представлення об'єктів адаптується для забезпечення пошуку у базах знань для аналізу RDF-графів у базах знань. У [23] підходи до вбудовування баз знань поділяються на дві основні групи: методи, засновані на перекладі, котрі інтерпретують зв'язки в базі знань як вектор трансляції між двома об'єктами, пов'язаними відношенням, і моделі семантичної відповідності, які використовують функції оцінки подібності об'єктів.

Приклади семантичних ІПС

Практично всі провідні промислові ІПС, такі як Google або Bing, так чи інакше реалізують семантичний пошук, але зазвичай не оприлюднюють детально методи, які використовують. Здебільшого вони підтримують пошук у масивах документів, що значно різняться рівнем структурованості та якістю метаданих, а бази знань використовують для вдосконалення запитів та для кращого фільтрування та сортування списку або повернутих документів у цьому контексті. Деякі семантичні ІПС (наприклад, Swoogle [24]) спеціалізуються на пошуку саме у базах знань, здебільшого поданих на основі стандартів Semantic Web – RDF та OWL, але використовують для цього традиційні пошукові алгоритми. Наприклад, SWSE [25] здійснює семантичний пошук у наборах RDF на основі їхніх метаданих та впорядковує результати пошуку з використанням алгоритму PageRank на графі відношень між URI та їхніми джерелами у тріплетах RDF.

SemSearch [26] здійснює пошук у ресурсах Semantic Web, перетворюючи запити користувача за ключовими словами на формальні запити. Sindice [27] підримує пошук у напівструктурованих даних великого обсягу як за ключовими словами та URI, так і структуровані запити.

Пошукова система Watson [28] здійснює пошук в онтологіях, виконуючи пошукові запити за ключовими словами та SPARQL-запити.

Напрямки семантизації пошуку

Наведений вище огляд показує, що методи семантичного пошуку можна розділити на дві основні групи залежно від цільового контенту [29] :

- методи підвищення релевантності класичних пошукових систем, де запит складається з тексту природною мовою (ПМ) – наприклад, списку ключових слів, а результати є ранжованим списком документів – наприклад, веб-сторінок або докумнетів;
- методи пошуку частково структурованих даних (зокрема, інформаційних об'єктів певної структури або RDF-трійок) у базі знань (наприклад, в онтології, семантично розміченому Wiki-ресурсі або графі знань) за запитом користувача, який може подаватися у формі ПМ-тексту або декларативної мови запитів, як-от, SPARQL.

Для обох груп використовується широкий спектр методів, таких як обробка природної мови для кращого розуміння запиту та контенту даних, технології Semantic Web для керування процесом пошуку з використанням декларативних баз знань, таких як онтології, а також машинного навчання.

Складові Інформаційно-пошукових систем

У найбільш загальному вигляді пошук складається з трьох складових: 1. запиту користувача q, що відображає його інформаційну потребу; 2. масиву даних I, в яких здійснюється пошук; 3. результатів пошуку R – тієї інформації, яку отримує користувач внаслідок виконання пошукової процедури. В такому випадку пошук можна розглядати як функцію $R = S(q, I)$, таку, що $R \subseteq I$.

Семантичний пошук є одним з підтипів інформаційного пошуку $R_{sem} \subseteq R$, який має ті самі складові, але доповнюється використанням зовнішніх джерел знань

К та методами їх застосування у пошуковому процесі: $R_{sem} = S_{sem}(q, I, K)$.

Методи пошуку S значною мірою залежать саме від цих трьох складових. У відкритому інформаційному просторі є можливість контролювати тільки q та S, тому дослідження та порівняння алгоритмів пошуку виконуються на спеціально сформованих тестових наборах I. Системи пошуку досить складно порівнювати саме через те, що вони значно різняться за всіма цими складовими. Тому доцільно визначати координати кожної застосовної системи у такому тривимірному просторі. Але для цього доцільно впорядкувати певним чином типові варіанти значень всіх цих параметрів відповідно до їхньої складності. Ця задача не є тривіальною через те, що серед них зустрічається багато непорівнюваних значень, і тому така класифікація є нечіткою. Тож виникає потреба проаналізувати можливості семантизації кожної із цих трьох складових та визначити, як саме вони впливають на методи пошуку.

Постановка задачі

Через велике розмаїття моделей, методів та засобів пошуку інформації, що ускладнюється внаслідок семантизації пошукових процедур, виникає проблема співставлення та вибору тих пошукових сервісів, що відповідають потребам користувачів застосовних систем. Цей вибір має враховувати як особливості ресурсів, серед яких планується здійснювати пошук, так і способи подання інформаційних потреб користувачів. Тому недостатньо аналізувати лише методи співставлення запитів з наявними джерелами інформації. Виникає потреба більш точно описувати властивості таких складових пошуку, як запити, результати запитів та інформаційні ресурси. Для цього пропонується тривимірна модель семантичного пошуку, яка базується на аналізі цих трьох складових та доповнює класифікацію систем семантичного пошуку. Для того, щоб обирати ІПС, що пертинентна певній задачі, необхідно визначити, які значення можуть мати ці параметри, та встановити часткове впорядкування цих значень там, де це можливо, відповідно до їхньої відповідності проблемі семантизації пошуку.

Пошукові запити та їх семанитизація

Запит користувача – це формалізований опис інформації, доступ до якої він прагне отримати. Цей опис може містити ключові слова, пов'язані логічними операторами; документ-зразок; тип документа і його тему за класифікатором; списки рекомендованих чи заборонених користувачем інформаційних джерел; обмеження стосовно часу або обсягу пошуку тощо. Деякі ІПС дають змогу також вводити такі параметри шуканого ІР, як час створення, обсяг, мова подання тощо. У більш складних або спеціалізованих пошукових механізмах користувач може вказувати тип інформаційного об'єкта, відомості про який він прагне отримати з наявних природномовних ІР (приміром, Web-сервіс, дані про особу чи організацію). Найпростішим варіантом запиту є непорожній набір послідовностей символів. Найчастіше це набір слів ПМ або чисел, але у більш узагальненому випадку можуть застосовуватися будь-які послідовності символів, що не потребують додаткової інтерпретації змісту (наприклад, пошук масок вірусів у файлах). Якщо ж використовуються саме слова певної мови – природної або формальної, то запит може уточнюватися та доповнюватися на основі знань щодо цієї мови.

Ускладнення запиту збільшує час його обробки, але використання елементів штучного інтелекту та менеджменту знань для побудови запитів дозволяють значно підвищити пертинентність його результатів. Тому дослідники в сфері СП значну увагу приділяють класифікації засобів семантизації пошукових запитів та доцільності їх застосування для різних задач.

Запити з ключових слів та їх розширення

Традиційні підходи до розширення запиту (query expansion – QE) спираються на інтеграцію неструктурованого корпусу та імовірнісних правил для виділення термінів – кандидатів для роз-

ширення. Ці методи не враховують семантику пошукового запиту, що призводить до неефективного пошуку інформації. Семантичні підходи до QE долають це обмеження, завдяки чому пошуковий запит розширюється значущими термінами, які відповідають потребам користувача. Ці підходи застосовують різні моделі та стратегії до різних структури знань – лінгвістичні методи, методи на основі онтології тощо. Таксономія таких методів, верхній рівень якої наведено на рис.1, пропонується в [30].

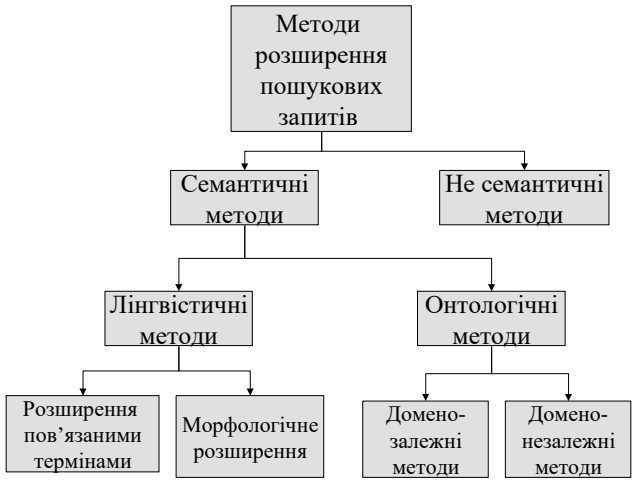

Рис.1. Таксономія методів розширення пошукових запитів

Основна задача ІПС полягає в тому, щоб відібрати документи, які містять потрібні користувачам відомості, та здобути з них ці відомості. Для цього ІПС обчислюють подібність між пошуковим запитом і документами та отримують список документів, розташованих у порядку зменшення подібності. Отриманий список документів іноді завеликий і містить багато нерелевантних документів. Основною проблемою, яка виникає під час пошуку, є невідповідність терміносистем користувачів та авторів документів: терміни, які автор використав для опису поняття в документі, відрізняються для користувачів. Основними причинами цього є вживання в ПМ слів зі схожим значенням (синонімія) та багатозначних слів (полісемія).

Ця проблема невідповідності ще більше посилюється короткими запитами, які користувачі застосовують для пошуку у Web: більшість таких запитів містить не більше двох-трьох слів [31], а цього недостатньо для автоматичного розв'язання неоднозначності.

Розширення запиту (QE) спрямоване на збільшення набору слів у запиті, і семантичні підходи мають суттєві переваги перед ручними та статистичними методами , оскільки вони розширюють кожен пошуковий запит значущими поняттями, які беруться зі структури знань (створеної вручну або автоматично) для представлення пошукового запиту. Нехай запит q містить непорожню множину ключових слів: $q = \{k_1,...,k_i \mid i \geq 1\}$, тоді розширення запиту – це перетворення його на набір $q_{ex} = \{k_1,...,k_m \mid m \geq i\}$. Залежно від того, з яких джерел здобуваються додаткові терміни та як саме здійснюється це здобуття, можна поділити методи розширення запитів на несемантичні (ручні або статистичні) та семантичні.

В свою чергу, існуючі семантичні підходи поділяють на лінгвістичні, онтологічні та змішані (гібридні). У лінгвістичних підходах значення слів виводяться з лінгвістичних баз знань – словників, тезаурусів тощо, які містять синоніми, гіпоніми та інші можливі зв'язки слів між поняттями, що відповідають словам з пошукового запиту, і є термінами розширення. Підходи, що базуються на онтологіях, використовують змістовні відношення між поняттями, що входять до запиту, щоб отримати значущі слова для розширення запиту. Змішаний підхід поєднує особливості лінгвістичного та онтологічного підходів: терміни розширення запиту можуть бути здобуті з баз знань різних типів.

Визначення розширення запиту на основі онтології. Нехай запит q містить непорожню множину ключових слів: $q = \{k_1,...,k_i \mid i \geq 1\}$, таких, що пов'язані з непорожнім набором понять $c = \{c_1,...,c_j \mid j \geq 1\}$ онтології O: $sem(O,q) = c$. Розширення запиту – це перетворення його на набір $q_{ex} = \{k_1,...,k_m \mid m \geq i\}$, таке, що це переформулювання запиту зберігає його семантику, тобто $sem(O,q_{ex}) = c$.

Таке визначення передбачає існування принаймні одного поняття для кожного ключового слова запиту . Крім того, кількість понять може не дорівнювати кількості ключових слів. Розширення запиту не означає розширення понять, передбачених у запиті (тобто мета запиту залишається незмінною), а саме розширення набору ключових слів через включення термінів, більш релевантних вже обраним поняттям, щоб ціль запиту стала більш конкретною та зрозумілою для ІПС.

Згідно з наведеним вище визначенням, можуть виникати два особливі випадки пошукового запиту: 1) всі ключові слова запиту стосуються одного поняття онтології, і тоді розширення запиту здійснюється на основі аналізу цього поняття в онтології [33]. ; 2) всі ключові слова запиту відповідають різним поняттям, тобто ключові слова можна вважати незалежними одне від одного.

Сутність e складається з непорожньої множини атомарних сутностей e^a: $e = \{e^a_1,...,e^a_n\}, n \geq 1$, де кожна атомарна сутність e^a відображає певний елемент інформації, який не можна розділити на інші сутності в обраному контексті. Тож, інтерпретація атомарності сутностей залежить від контексту. Наприклад, ім'я людини в одному контексті може розглядатися як атомарна сутність, а в іншому контексті розкладатися на ім'я та прізвище.

Документ D з точки зору пошуку – це набір сутностей, $D = \{e_1,...,e_k\}, k \geq 1$. В такому розумінні сам документ не є сутністю, але містить набір сутностей. Предметна область (ПрО) – це підмножина світу, що характеризується певною множиною знань, яка може описуватися через корпус ПрО $K = \{D_1,...,D_m\}, m \geq 1$. Кожна сутність ПрО може міститися в документах більш ніж один раз.

Експерти в певній ПрО використовують у побудові запитів власні знання і відомі їм терміни, а для запитів в областях за межами сфери їхньої компетенції такі досвідчені користувачі широко використовували тезауруси та інші зовнішні джерела знань для знаходження термінів. Знання ПрО впливає на поведінку користувачів та забезпечує більш ефективні стратегії вибору термінів, коротші запити [34] і зменшення помилок у тактиці пошуку [35].

Але користувачі-початківці частіше використовують лише свої обмежені знання про область пошуку і рідко звертаються до інших джерел знань, хоча низький рівень їхніх знань потребує більше змін початкового запиту для отримання потрібної інформації. Тому саме вони потребують автоматизованих засобів уточнення та вдосконалення їхніх запитів з використанням знань ПрО, зробивши встановлення зв'язків між запитами та документами більш коректним.

Багато досліджень показує, що онтології дозволяють подолати розрив між термінами запиту та документами, використовуючи семантику ПрО. Онтології та тезауруси, які можуть розглядатися як окремі випадки онтологій зі спрощеною формалізацією, можуть використовуватися і для розширення запиту як джерело релевантних термінів, і у обробці його результатів для усунення неоднозначності та обчислення подібності між запитами та документами.

Онтологія ПрО може розглядатися як комбінація інтенсіональних і екстенсіональних знань. Інтенсіональні знання про домен (TBox) подібні до схеми бази даних та формалізують структуру об'єктів ПрО як набір аксіом, а екстенсіональні знання (ABox) відображають відомості про екземпляри об'єктів. Інтенсіональне знання виражається у TBox.

Основна мета розширення запиту полягає в тому, щоб обчислити терміни, які відповідають намірам користувача, але не містяться в його запиті, і додати їх до початкового пошукового запиту. Традиційні підходи використовують статистичний аналіз вмісту корпусу текстів для знаходження термінів-кандидатів. Тому такі підходи добре працюють тільки тоді, коли доступний великий корпус, а вміст цього корпусу релевантний ПрО пошукового запиту. Семантичні підходи не мають таких обмежень, оскільки вони базуються на незалежних від корпусу зовнішніх джерелах знань (наприклад, лексичному тезаурусі або онтології ПрО).

Онтологія містить знання про структуру понять ПрО, тобто являє собою потенційне джерело відомостей щодо семантично пов'язаних термінів. Семантичне розширення запиту забезпечує інтерпретацію пошукового запиту, використовуючи інформацію про структуру понять. Терміни розширення отримують на основі визначення кількісних оцінок семантичної подібності між початковими термінами пошукового запиту та іншими поняттями ПрО: запит доповнюється тими термінами, які найближче до термінів у запиті користувача.

Структура знань може бути пов'язаною з ПрО (тобто описувати класифікацію та структуру об'єктів певної області) або загальною (наприклад, Cyc і EuroWordNet). Поняття, відношення між поняттями та властивості понять становлять словник структури знань, тим самим фіксуючи набір семантично значущих термінів для розширення запитів. Тому ефективність такого розширення значною мірою залежить як від якості словникового запасу (його точності, повноти, актуального представлення знань), так і пертинентності обраної структури знань інформаційним потребам користувача та відповідності рівня її узагальненості та складності.

Прості таксономічні зв'язки , такі як гіпернімія (гіпернім (hypernym) – "Has-A" – слово з широким значенням, під яке підпадають більш конкретні слова, такі як, "тварина" – це гіпернім слова "собака") і гіпонімія (гіпонім (hyponym) – "Is-A" – слово з конкретнішим значенням, ніж більш загальний термін, наприклад, "пацюк" – це гіпонім до слова "тварина"), дозволяють переходити таксономію вгору і вниз для більш загальних категорій і підкатегорій відповідно. Використання таких зв'язків для розширення запитів забезпечує отримання більш загальних або більш конкретних понять для термінів пошукового запиту зі структури знань. Однак вибір відповідної ієрархічної відстані (наприклад, два або більше рівнів від вихідного поняття) для отримання понять-кандидатів розширення зі структури знань залишається досить складною проблемою.

Інший тип підходів до розширення запитів зосереджується на нетаксономічних відношеннях структури знань, таких як синонімія, тропонімія, антонімія, відношення "частина-ціле", семантична роль, залежність, типове розташування, причинно-наслідкові відношення тощо [36], забезпечуючи структурне представлення змісту слів. Наприклад, у [37] запропоновано метод структурних семантичних взаємозв'язків (structural semantic interconnections – SSI), який створює структурні специфікації можливих значень для кожного слова в контексті та вибирає найкращу гіпотезу відповідно до граматики, що описує зв'язки між змістовними специфікаціями. Метод може застосовуватися до проблем семантичного усунення неоднозначності, таких як автоматична побудова онтології, семантичне розширення запитів та усунення неоднозначності слів у глосарію.

Однорівневі відношення, такі як синоніми, антоніми та зв'язки пов'язаних понять, є ефективними для усунення неоднозначності у значеннях термінів запиту та можуть бути легко отримані з лінгвістичних джерел знань та онтологій ПрО (наприклад, словників, тезаурусів або WordNet).

Значення слів, що описані в лінгвістичних базах знань (наприклад, у WordNet), широко використовуються багатьма дослідниками для усунення неоднозначності початкових термінів пошукового запиту [38] : слова, які здобуті зі зв'язків глосарію WordNet, є кращими кандидатами для розширення запитів, ніж слова вищого або нижчого рівня таксономії.

Кожну категорію можна далі розділити на підкатегорії відповідно до ключових характеристик. Лінгвістичні підходи базуються на інформації про властивості природної мови для створення термінів розширення. До них належать морфологічні підходи, методи на основі синонімії.

Підходи морфологічного розширення використовують морфологічні форми слів запиту (наприклад, основу, частину мови та форми слова) для створення функцій розширення. Експерименти з використанням корпусів різних мов продемонстрували, що розширення запитів морфологічними варіантами термінів за-

питу (автоматично здобутих з документів) дає задовільну продуктивність пошуку [39].

Підходи до розширення з використанням пов'язаних термінів використовують синонімію та інші типи семантично пов'язаних слів природної мови для розширення пошукового запиту. Джерелами таких знань є словники та тезауруси.

Наприклад, найбільш відома лексична база даних WordNet [6] об'єднує функції словника та тезауруса. Вона класифікує слова ПМ на іменники, прикметники, дієслова та прислівники, а також групує слова, які мають однакове значення, в набори, що називаються синсетами. Кожен синсет має семантичні зв'язки з іншими (наприклад, зв'язки гіпонімів і меронімів). Саме синсети надають інформацію для розширення запитів: синсети, найбільш подібні до ключових слів запиту, додаються до цього запиту.

В онтологічних підходах до розширення запитів поняття з онтології додаються до початкових запитів. Для цього можуть використовуватися як онтології верхнього рівня (доменно-незалежні), так і онтології окремих ПрО, а також більш специфічні онтології різної виразності – онтології задач, користувачів тощо. Для здобуття понять-кандидатів можуть використовуватися запити мовою SPARQL. Якщо пошук здійснюється в певній ПрО, то доцільно застосовувати релевантні онтології. У більш узагальнених випадках використовують доменно-незалежні онтології, такі як OpenCyc [40], YAGO [41], DBpedia [42] і UNIpedia [43]. Особливий інтерес становлять онтології, пов'язані із Wiki-технологіями, тому що користувачам легше сприймати їхню структуру та обсяг.

Наприклад, у [44] запропонована модель збагачення семантичного запиту з використанням онтологій Wikipedia та Dbpedia для отримання термінів для розширення, що семантично споріднені з ключовими словами запиту. Але використання таких доменно-незалежних онтологій призводить до двох проблем: 1. загальні онтології містять неоднозначні терміни, що мають різні значення у різних ПрО; 2. такі онтології зазвичай не містять спеціалізовані властивості та специфічні терміни окремих ПрО.

Звичайні методи семантичного розширення запитів не використовують контекст пошуку окремого користувача (тобто профіль користувача чи історію пошуку), необхідний для визначення правильного контексту запиту користувача. Але визначення контексту пошуку запиту користувача важливо з двох причин: 1) однакові пошукові запити різних користувачів можуть мати різні цілі; 2) інформаційні потреби одного користувача можуть з часом змінюватися. Таким чином, потрібно персоніфікувати розширення запитів та розробити засоби відбору актуального контексту.

Окрім профілю користувача та історії його запитів, джерелами для збору персоналізованої інформації можуть бути його профілі та поведінка у соціальних мережах, такі як Twitter, Facebook і Linkedln [45]. Але потрібно враховувати, що здебільшого така інформація є закритою та охороняється законами про персональні дані.

Мультионтологічний підхід, що полягає у використанні кількох онтологій до розширення запитів, є ефективним інструментом для пошуку на перетині кількох ПрО [46]. Але його застосування значно ускладнює необхідність узгодження та вирівнювання таких онтологій.

Тож, запити з ключових слів (незалежно від того, були ці ключові слова введені самим користувачем, чи отримані завдяки різноманітним методам розширення запитів – у тому числі й семантичним) з точки зору ІПС обробляються однаково.

Природномовні запити

Багато сучасних ІПС забезпечують користувачам можливість формулювати запити природною мовою. Обробка природномовних пошукових запитів здебільшого стосується перетворення ПМ-конструкцій у структуровані запити з використанням методів морфологічного, лінгвістичного та семантичного аналізу [47], що знаходяться поза сферою даного дослідження.

Обробка запитів ПМ передбачає такі функції, як видалення стоп-слів, морфологічний пошук (відображення слів запиту у базову форму), розпізнавання частин мови.

Багато ІПС, що підтримують ПМ-запити, використовують онтології для уточнення та співставлення елементів запиту з поняттями відповідної області [48]. Якщо користувач ставить запитання, тобто вводить набір слів, що починаються з прислівника (“який”, “коли”, “як” тощо), які потрібно інтерпретувати у структурований запит, побудувавши відповідну логічну форму: наприклад, перетворити “хто” на “категорія:персоналія”. Якщо є наявні знання щодо ПрО, тоді може бути виконане подальше перетворення, що відповідає специфіці ПрО. Наприклад, перетворити “категорія:персоналія” на “категорія:працівник” або “категорія:пацієнт”. Далі логічна форма перетворюється на вираз відповідної ІПМ. Інформаційними ресурсами, які використовуються для відповідей на запити, можуть бути зовнішні або внутрішні бази знань та онтології. У найбільш узагальненому вигляді обробка ПМ-запитів з використанням онтологій у пошукових системах наведена на рис.2.

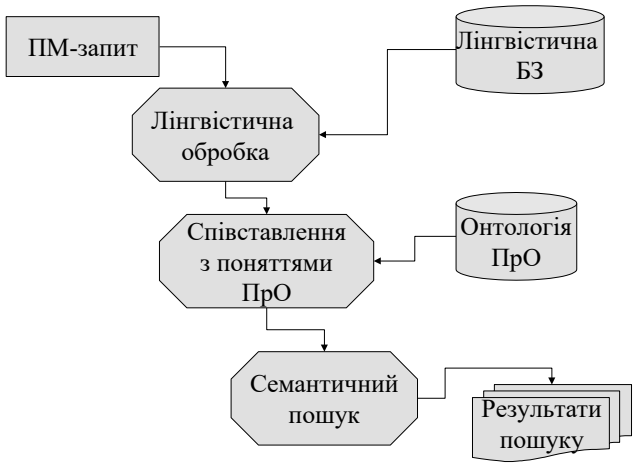

Рис.2. Обробка ПМ-запитів з використанням онтологій

Приклади ІПС, що обробляють ПМ-запити – SemanticWeb Search Engine (SWSE) [49] та Orakel [50]. Google також використовує підходи на основі ПМ для обробки запитів.

Структуровані запити

Більш складним варіантом пошукових запитів є структуровані запити, які дозволяють користувачам формально описувати умови до відомостей, які вони хочуть знайти. Багато традиційних ІПС підтримують такі прості елементи структурування, як кон’юнкція та диз’юнкція ключових слів [51]

Для побудови таких запитів використовуються спеціальні формальні мови – *інформаційно-пошукові мови* (ІПМ) – спеціальні формалізовані штучні мови, створені для відображення інформаційної потреби користувача у такій формі, що забезпечує її співставлення з інформацією про наявні ІР. Залежно від методу побудови системи пошуку ІПМ поділяють на класифікаційні та дескрипторні [52] (рис.3).

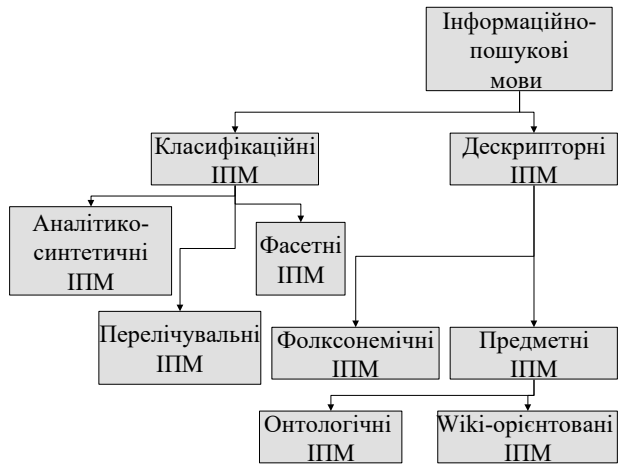

Рис.3. Класифікація ІПМ

Класифікаційні ІПМ порівнюють об’єкти за наборами ознак (які можуть бути пов’язані родо-видовими відношеннями), щоб віднести кожен об’єкт до певного класу. До таких ІПМ належать перелічувальні, аналітико-синтетичні та фасетні мови.

Перелічувальні ІПМ використовують ієрархічні набори ознак пронумерованих класів. На верхньому рівні такі класифікації містять найбільш загальну ознаку. Приклади – десяткова класифікація Дьюї, бібліотечні класифікатори.

Фасетні ІПМ містять сукупності фасетів («фасетна формула» пошукового запиту), які описують комбінації спільних оз-

нак об'єктів. Приклад – класифікація двокрапкою Шіалі Рамамріта Ранганатаном.

Аналітико-синтетичні ІПМ поділяють об'єкти на класи за незалежними ознаками від загальних до більш конкретних. Приклад – Універсальна десяткова класифікація (УДК), де перший фасет є головним, а інші – допоміжні [53] .

Дескрипторні ІПМ описують запит та об'єкти за допомогою ключових слів. Ключове слово, що виражає найзагальніше, головне значення, за допомогою якого можна точно описати зміст документу або запиту, називається дескриптором. Упорядковані в алфавітному порядку дескриптори та їхні синоніми утворюють дескрипторний словник, тоді як більш складні зв'язки між дескрипторами та їхніми значеннями відображає інформаційно-пошуковий тезаурус – структурований словник, що формалізує семантичні відношення (такі як відношення еквівалентності, ієрархічні та асоціативні) між термінами ПМ. Це дозволяє встановлювати змістовні зв'язки між тими ключовими словами запиту, що не є дескрипторами, та власне дескрипторами.

Серед ІПМ, що використовуються у веб-орієнтованих системах, виділяють такі підкатегорії [54]:

- класифікаційні ІПМ: пошук виконується на основі певної класифікаційної системи, каталогів або таксономії (наприклад, пошук на основі категорій у Вікіпедії);
- предметні ІПМ: пошук виконується за допомогою ключових слів або певних предметних рубрик (наприклад, семантичний пошук у ресурсах на основі Semantic MediaWiki [55]);
- дескрипторні ІПМ: пошук виконується за допомогою дескрипторів;
- фолксонемічні ІПМ: у пошуку використовуються різноманітні типи фолксономій, що візуалізуються як хмари тегів, глосаріїв та онтологій.

У багатьох електронних бібліотеках та інформаційно-аналітичних порталах підтримуються одночасно кілька видів пошукових сервісів, що базуються на різних типах ІПМ.

У запитах до структурованих та слабо структурованих ІР можуть застосовуватися складніші ІПМ. Основним питанням у кожній ІПМ є складність оцінки запиту і, зокрема, вплив кожного компонента мови на цю складність. Чим більш розгалуженою є структура запиту – тим більша точність і тим кращий результат пошуку. З іншого боку, ускладнення структури запиту призводить до двох негативних наслідків: ускладнення самого процесу побудови запиту для користувачів та зростання обсягу метаданих, які потрібно зберігати та обробляти по кожному документу.

Структуровані запити передбачають використання знань щодо структури об'єктів пошуку. Для того, щоб задавати у запиті певні умови щодо властивостей (як формальних, так і семантичних) цілого документу або його фрагментів, потрібно визначити назви цих елементів, тобто використовувати певну схему метаданих або структуру того об'єкта, інформацію про який потрібно знайти.

Існує багато мов запитів для пошуку в RDF, такі як DQL, N3QL, R-DEVICE, RDFQ, RDQ, RDQL, SeRQL і т.д., але найпоширенішою є SPARQL – стандарт W3C, який, на відміну від SQL з неоднозначною граматикою і семантикою, має чітку структуру і більшу виразність [55]. Основна частина запиту на SPARQL – шаблон, що описує підграф, який потрібно знайти в графі RDF. Цей шаблон представляється у вигляді набору трійок з перемінними. На сьогодні SPARQL є однією з найбільш виразних мов обробки даних. Крім мови запитів, стандарт SPARQL регламентує протокол взаємодії з базою даних і формат результату, що є великим кроком вперед порівняно із SQL. Наприклад, для пошуку в онтологіях використовують запити мовою SPARQL [56]. Це мова, розроблена для моделі даних RDF. Використання твердження SPARQL як стандартної мови запитів для RDF дозволяє багатьом сховищам даних стати точками доступу SPARQL, у такий спосіб забезпечуючи гнучкий обмін даними між системами. Ця мова є елементом стеку технологій Semantic Web, що підтримує витяг значень зі структурованих і напівструктурованих даних, дослідження відношень між даними та складні об'єднання розрізнених баз даних в одному запиті.

Поширеним прикладом частково структурованих ІР є різноманітні семантичні розширення Wiki-ресурсів, в яких елементи контенту явно пов'язуються за допомогою розмітки з поняттями певної ПрО. Пошук у таких ІР підтримується відповідними ІПМ, що враховують засоби та можливості такої семантизації даних. Виразність ІПМ, що використовується у семантичних Wiki, значно менша за виразність SPARQL, тому що виразність засобів подання знань у Wiki-ресурсах також значно поступається виразності RDF та OWL.

KiWi (http://www.kiwi-project.eu/) – це семантичне розширення Wiki-технології з додатковими можливостями здобуття інформації, персоналізації, логічного виведення та створення запитів. Основними одиницями інформації в KiWi є елементи контенту (Content Items), що розширюють концепцію Wiki-сторінок і можуть бути вкладеними. Кожен такий елемент однозначно ідентифікований своїм URI, може містити фрагменти тексту або мультимедіа, посилання та теги [57] . KWQL – це мова запитів на основі правил, яка поєднує характеристики пошуку за ключовими словами з характеристиками веб-запитів для уможлення різноманітних запитів у KiWi. Мова дозволяє створювати комбіновані запити щодо текстового вмісту, метаданих, структури документа та формальних семантичних анотацій. Запити KWQL варіюються від елементарних і відносно неспецифічних до вибору складних і повністю визначених метаданих.

Для пошуку у Wiki-ресурсах, що семантизовані на основі Semantic MediaWiki, використовується проста, але потужна мова запитів SMW-QL [58]. Мова запитів SMW-QL дозволяє фільтрувати сторінки за заданими критеріями і виводити як результати запиту тільки потрібну інформацію, а не весь текст Wiki-сторінки. Якщо сторінки, з яких отримуються потрібні дані, будуть змінюватися, то результати запитів також будуть автоматично оновлюватися, забезпечуючи несуперечність і погодженість даних.

Найчастіше використовуються вбудовані запити, сполучені з функцією ask, яка має три основні параметри: перший параметр задає умови щодо набору категорій та значень семантичних властивостей сторінок; другий параметр визначає, які саме значення семантичних властивостей цих сторінок потрібні користувачу, а третій параметр вказує форму подання результатів. Таким чином користувач може отримати не тільки перелік документів, а саме потрібні елементи їхнього контенту [59].

Тож, на основі аналізу ІПМ, можна виділити наступні типи запитів, що обробляються в ІПС:

- набори ключових слів (що безпосередньо вводяться користувачами чи будуються на основі таких запитів за допомогою різних методів розширення);
- ПМ-запити, в яких значення має також порядок слів та їхня форма (в ІПС такі запити також перетворюються на набори ключових слів, але вибір ПМ та методи перетворення значною мірою впливають на результати пошуку);
- структуровані запити, в яких явно описані логічні відношення (диз'юнкція, кон'юнкція, заперечення тощо) між термінами та умови щодо властивостей.

Відповідно ІПС за функціоналом обробки пошукових запитів можна класифікувати за наявністю наступних сервісів:

- співставлення набору ключових слів (довільних послідовностей символів) з наявними ІР – мінімальний функціонал ІПС;
- розширення набору ключових слів за допомогою зовнішніх та внутрішніх джерел знань;
- перетворення ПМ-запитів на набори ключових слів (видалення роздільників, ключових слів, виправлення орфографічних помилок, перетворення слів на нормальну форму);
- обробка структурованих запитів, де ключові слова пов'язані логічними відношеннями та обмеженнями;
- перетворення ПМ-запитів на структуровані запити на основі морфологічного, лінгвістичного та семантичного аналізу.

Інформаційні ресурси, серед яких здійснюється пошук

Інформаційні ресурси, серед яких здійснюється пошук різними ІПС, значно різняться [60]:

- моделями подання інформації;
- рівнем структурованості контенту;
- ступенем розподіленості;
- обсягом.

Пошук може здійснюватися на окремому носії, на певному сайті або порталі, у локальній мережі, у базі знань, у відкритому середовищі Web тощо. Інформація, серед якої здійснюється пошук, може бути однорідною або гетерогенною. Метадані, що характеризують наявну інформацію, можуть бути уніфіковані або різнорідні та потребувати інтеграції й узгодження. Чим більше попередніх умов накладено на структуру та подання відомостей в ІР, тим складніші та точніші пошукові запити можна будувати з використанням цих вимог.

Залежно від рівня структурованості, ІР поділяють на:

- структуровані;
- слабо структуровані;
- неструктуровані.

ІР можуть розглядатися як неструктуровані, якщо вони містять певні структурні елементи, але ці елементи не можуть бути використані для мети пошуку [55].

Найбільш розповсюдженою моделлю збереження структурованих даних з кінця 70-х років 20 ст. є реляційна модель, а стандартом на їхню обробку – мова SQL. Однак для НСД ця модель неефективна.

Існує велика кількість ІПС, що спеціалізуються на пошуку певних типів ІР зі специфічними метаописами (відео, музика, мапи, книги тощо) або на пошуку у певних ПрО (наприклад, товари в електронних магазинах). Крім того, у багатьох інформаційно-аналітичних системах використовуються спеціалізовані сервіси, що підтримують пошук різноманітних складних інформаційних об’єктів – подорожей, навчальних курсів [61].

Якщо пошук здійснюється в ІР великого обсягу або таких, що швидко змінюються, то це потребує застосування масштабованих технологій подання даних та відповідних методів пошуку в них. Наприклад, для задачі, що виходять за рамки реляційної моделі, прийнято використовувати моделі даних класу NoSQL, такі як документо-орієнтовані, об'єктні та графові БД. Такі БД мають певні обмеження на операції, що підтримуються традиційними БД. Наприклад, великі розподілені БД повністю відмовляються від транзакцій, що забезпечує підвищення продуктивності за рахунок використання паралелізму. Інший клас задач, які важко розв'язувати на реляційній моделі, – це задачі на сильно зв'язаних даних (графові задачі). Для них сьогодні найбільше поширення мають RDF-сховища, які використовують стандарти W3C для мови RDF (Resource Description Framework) і запити SPARQL.

Ще однією важливою умовою пошуку є відкритість даних. Наявність доступу до даних є основою їх повторного використання.

Багато вимог щодо підтримки доступності та ефективного пошуку інформації відображено в FAIR [62] – принципах керування даними (а саме – знаходжуваності, доступності, інтероперабельності та повторного використання) без утручання користувача, що були розроблені для формування цифрової інфраструктури трансферу наукових даних. Згідно FAIR, функції пошуку, здобуття і представлення даних реалізують не користувачі, а інформаційна система. Водночас мова йде не тільки про власне дані і метадані, а й про алгоритми та інструменти керування ними. Щоб використовувати дані, їх необхідно спочатку знайти там, де вони зберігаються. Метадані та дані повинні бути легко доступними як для людей, так і для комп’ютерів, і тому вимоги FAIR чітко характеризують ті властивості ІР, що мають забезпечити їх знаходження та автоматизовану обробку метаданих.

Значна частка існуючих ІР, що розроблялися незалежно до цих принципів, але з урахуванням можливостей пошуку, відповідають вимогам FAIR. Наприклад, наведений в [63] аналіз виразних властивостей середовища Semantic MediaWiki свідчить про те, що семантичні Wiki-

ресурси, які будуються в цьому середовищі, відповідають вимогам до відкритих даних великого обсягу.

Результати пошуку

Серед типів пошуку виокремлюють:

- адресний пошук, коли результатом структурованого запиту є посилання (адреси, імена) документів, файлів, вебсайтів тощо;
- документальний пошук, коли результат запиту – це або сам документ, або додаткові метадані про нього;
- фактографічний пошук, коли результатом пошуку є певна інформація, здобута з доступних ІР.

Залежно від того, як задаються умови пошуку, результати пошуку можуть обмежуватися певною кількістю знайдених об'єктів або певною межею, що визначає рівень релевантності запиту та тих об'єктів, з якими цей запит співставляється.

Усі ці типи пошуку можуть бути результатом обробки як набору ключових слів, так і структурованого запиту, але в першому випадку потрібно окремо вказувати, що саме має бути результатом запиту.

ІР, серед яких здійснюється пошук, можуть значно різнитися (містити текст, зображення, відео, програмний код, структуровані дані тощо) та супроводжуватися різними видами метаданих (що характеризують документ в цілому або також ї його складові), і саме це є причиною того, що й результати пошуку можуть вказувати на певні документи або знаходити окремі елементи цих документів, що відповідають запиту. Крім того, запити можуть явно визначати умови щодо того, яку інформацію потрібно надати користувачу.Тому іноді структурно прості результати пошуку можуть бути результатом семантичного пошуку та обробки запитів зі складною структурою.

Отже, за конкретизацією результати запитів можна класифікувати наступним чином:

- бінарні ("так-ні") відповіді щодо наявності потрібної інформації у наявних ресурсах (наприклад, чи наявні документи, що містять рядок символів "абв" або чи існує сайт "abc.org");
- кількісні (скільки документів містять рядок символів "абв" або скільки разів у поточному документі зустрічається цей рядок);
- посилання на документи (URL, імена файлів, вікісторінки тощо), які відповідають умовам запиту;
- посилання на інформаційні об'єкти (наприклад, класи онтології або елементи документів), які відповідають умовам запиту;
- обрані відповідно до умов запиту значення властивостей знайдених об'єктів (документів або їхніх елементів), які визначені користувачем;
- більш складні результати обробки таких знайдених значень властивостей (наприклад, сума отриманих значень або графік).

Досить часто підсистеми семантичного пошуку підтримують усі ці варіанти надання результатів пошуку (наприклад, пошук у семантизованих вікіресурсах), але наявність такої класифікації може значно спростити аналіз придатності конкретного технологічного середовища для задач користувача

Висновки та перспективи

Основною метою цього дослідження було визначення тенденцій розвитку сервісів семантичного пошуку, що можуть бути застосовані для підтримки функціювання інформаційно-аналітичних порталів, які базуються на вікітехнологіях [64]. Аналіз існуючих підходів до семантизації пошуку та результатів їх застосування дозволяє виокремити перспективні напрямки впровадження елементів онтологічного аналізу у розробку таких систем.

Запропонована тривимірна модель семантичного пошуку пропонується як додатковий інструмент опису та співставлення пошукових систем, що використовують різноманітні елементи штучного інтелекту та менеджменту знань для більш ефективного та пертинентного задоволення інформаційних потреб користувачів. Як такі методи виконання співставлення між

запитами та ІР в цій моделі не аналізуються, тому що це співставлення є наступним кроком виконання пошуку. Потрібно зауважити, що значення параметрів, які аналізує ця модель, не є взаємозаперечними, тобто та сама ІПС може підтримувати кілька варіантів пошуку. Крім того, засоби подання запитів та ресурсів не завжди є порівнюваними (наприклад, структуровані дані, що описуються різними схемами метаданих, можуть бути орієнтовані на різні типи задач та відображати різні аспекти даних, але водночас одна схема не є повнішою, але виразнішою за іншу). Так само, різні способи фільтрації та подання результатів мають відповідати різним потребам і не завжди можуть порівнюватися за виразністю (наприклад, можливість візуалізації отриманих значень у вигляді графіка не може бути порівняна з можливістю виконання логічних або арифметичних операцій над цими значеннями). Але наявність самих критеріїв порівняння та розширений набір їхніх параметрів надає більш зручний апарат для вибору відповідної ІПС. Отже, така модель дозволяє виявляти ІПС, для яких перетинаються тріади "запит-ІР-результат" та виконувати їх порівняння саме на цих підкласах пошукових задач. Це дозволяє визначати, які алгоритми пошуку виявляються більш пертинентними для конкретних задач користувачів і на основі цього обирати такі сервіси як джерело інформації для подальшої обробки.

Важливою особливістю запропонованої моделі є те, що вона використовує лише ті характеристики ІПС, які можуть бути проаналізовані користувачами (алгоритми, які використовуються в ІПС для співставлення запитів та ресурсів, доступні тільки розробникам цих систем, а їхні наявні описи – приміром, в наукових публікаціях або в документації – можуть значно відрізнятися від використаних у поточній версії програмної реалізації).

References

1. Rogushina, J. (2015) The Web semantic ontology-based search: development of models, tools and methods – Melitopol, 291 p. (in Ukrainian)
2. Bast, H., Buchhold, B, Haussmann, E. (2016) Semantic search on text and knowledge bases. Foundations and Trends in Information Retrieval 10(2-3): 119-271.
3. Mangold, C. (2007) A survey and classification of semantic search approaches. Metadata Semantic Ontologies 2(l):23-34.
4. Manning, C. (2011) Part-of-speech tagging from 97% to 100%: Is it time for some linguistics? Gelbukh AF (Computational Linguistics and Intelligent Text Processing, Springer Berlin Heidelberg, 171-189.
5. Hua, W., Wang, Z., Wang, H, Zheng, K, Zhou, X (2015) Short text understanding through lexical-semantic analysis. In: 2015 IEEE 31st International Conference on Data Engineering, 495-506.
6. Fellbaum, C. (2010). WordNet. In: Theory and applications of ontology: computer applications, 231-243.
7. Pehcevski, J., Vercoustre, A., Thom, J. (2008) Exploiting locality of Wikipedia links in entity ranking. In: Advances in Information Retrieval, Springer Berlin Heidelberg, , 258-269.
8. Kaptein, R., Serdyukov, P., de Vries A., Kamps, J. (2010) Entity ranking using wikipedia as a pivot. In: Proc. of the 19th ACM Conference on Information and Knowledge Management, CIKM 2010, 69-78.
9. Schuhmacher, M., Dietz, L., Ponzetto S (2015) Ranking entities for web queries through text and knowledge. In: Proc. of the 24th ACM International on Conference on Information and Knowledge Management, 1461-1470.
10. Tran, T., Cimiano, P., Rudolph, S., Studer, R. (2007) Ontology-based interpretation of keywords for semantic search. In: Proc. of the 6th International The Semantic Web and 2Nd Asian Conference on Asian Semantic Web Conference, ISWC'07/ASWC'07, 523-536.
11. Schuhmacher, M., Ponzetto, S.P. (2013) Exploiting dbpedia for web search results clustering. In: Proc. of the 2013 Workshop on Automated Knowledge Base Construction, ACM, DOI 10.1145/2509558. 2509574.
12. Mikolov, T., Chen, K., Corrado, G., Dean, J. (2013) Efficient estimation of

word representations in vector space. arXiv preprint arXiv:1301.3781.
13. Zou, X. (2020). A survey on application of knowledge graph. In: Journal of Physics: Conference Series Vol. 1487, No. 1, 012-016.
14. Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P. N., Bizer, C. (2015). Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. Semantic web, 6(2), 167-195.
15. Vrandečić, D., Krötzsch, M. (2014). Wikidata: a free collaborative knowledgebase. Communications of the ACM, 57(10), 78-85.
16. Horrocks, I., Tessaris, S. (2002) Querying the semantic web: A formal approach. In: Horrocks I., Hendler J. (eds) The Semantic Web, ISWC 2002, 177-191
17. Stojanovic, N., Studer, R., Stojanovic, L. (2003). An approach for the ranking of query results in the semantic web. In: The Semantic Web-ISWC 2003: Second International Semantic Web Conference, . Proc. 2, 500-516.
18. Maedche, A., Motik, B., Stojanovic, L., Studer, R., Volz, R. (2003). An infrastructure for searching, reusing and evolving distributed ontologies. In: Proc. of the 12th international conference on World Wide Web, 439-448).
19. Tonon, A., Demartini, G., Cudrd-Mauroux, P. (2012) Combining inverted indices and structured search for ad-hoc object retrieval. In: Proc. of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '12,125-134, DOI 10.1145/2348283
20. Pound, J., Mika, P., Zaragoza, H. (2010). Ad-hoc object retrieval in the web of data. In: Proc. of the 19th international conference on World Wide Web, 771-780.
21. Rocha, C., Schwabe, D., Aragao, M. P. (2004). A hybrid approach for searching in the semantic web. In Proc. of the 13th international conference on World Wide Web, 374-383).
22. Zhang, L., Yu, Y., Zhou, J., Lin, C., & Yang, Y. (2005). An enhanced model for searching in semantic portals. In Proc. of the 14th international conference on World Wide Web, 453-462).
23. Wang, Q., Mao, Z., Wang, B., Guo, L. (2017) Knowledge graph embedding: A survey of approaches and applications. In: IEEE Transactions on Knowledge and Data Engineering 29(12):2724-2743,
24. Ding, L., Finin, T., Joshi, A., Pan, R., Cost, R. S., Peng, Y., Sachs, J. (2004). Swoogle: a search and metadata engine for the semantic web. In: Proc. of the thirteenth ACM international conference on Information and knowledge management, 652-659.
25. Hogan, A., Harth, A., Umbrich, J., Kinsella, S., Polleres, A., Decker, S. (2011) Searching and browsing linked data with swse: The semantic web search engine. In: Web Semantics: Science, Services and Agents on the World Wide Web 9(4):365-401.
26. Lei, Y., Uren, V.S., Motta, E. (2006) Sem-search: A search engine for the semantic web. In: Managing Knowledge in a World of Networks, 15th International Conference EKAW-2006, 238-245.
27. Oren, E., Delbru, R., Catasta, M., Cyganiak, R., Stenzhom, H., Tummarello, G. (2008) Sindice.com: a document-oriented lookup index for open linked data. In: IJMSO 3(1):37-52.
28. d'Aquin, M., Motta, E. (2011) Watson, more than a semantic web search engine. In: Semantic web 2(l):55-63.
29. Cudré-Mauroux, P. (2019). Semantic Search. https://exascale.info/assets/pdf/cudre2018abigdata.pdf.
30. Raza, M. A., Mokhtar, R., Ahmad, N., Pasha, M., Pasha, U. (2019). A taxonomy and survey of semantic approaches for query expansion. In: IEEE Access, 7, 17823-17833.
31. Gabrilovich, E., Broder, A., Fontoura, M., Joshi, A., Josifovski, V., Riedel, L., Zhang, T. (2009). Classifying search queries using the web as a source of knowledge. In: ACM Transactions on the Web (TWEB), 3(2), 1-28.
32. Wu, J., Ilyas, I., Weddell, G. (2011). A study of ontology-based query expansion. In: Technical report CS-2011–04. https://cs.uwaterloo.ca/research/tr/2011/CS-2011-04.pdf.
33. Qiu, Y., & Frei, H. P. (1993). Concept based query expansion. In: Proc. of the 16th annual international ACM SIGIR

conference on Research and development in information retrieval, 160-169.
34. Duggan, G. B., Payne, S. J. (2008). Knowledge in the head and on the web: Using topic expertise to aid search. In: Proc. of the SIGCHI conference on Human factors in computing systems, 39-48.
35. Wildemuth, B. M. (2004). The effects of domain knowledge on search tactic formulation. In: Journal of the american society for information science and technology, 55(3), 246-258.
36. Loukachevitch, N. V., Dobrov, B. V. (2004). Development of Ontologies with Minimal Set of Conceptual Relations. In: LREC.
37. Navigli, R., Velardi, P. (2004). Learning domain ontologies from document warehouses and dedicated web sites. In: Computational Linguistics, 30(2), 151-179.
38. Liu, S., Liu, F., Yu, C., Meng, W. (2004). An effective approach to document retrieval via utilizing WordNet and recognizing phrases. In: Proc. of the 27th annual international ACM SIGIR conference on Research and development in information retrieval, 266-272.
39. Moreau, F., Claveau, V., Sébillot, P. (2007). Automatic morphological query expansion using analogy-based machine learning. In: Advances in Information Retrieval: 29th European Conference on IR Research, ECIR 2007, Proc. 29, 222-233).
40. Best, B. J., Gerhart, N., Lebiere, C. (2010). Extracting the ontological structure of OpenCyc for reuse and portability of cognitive models. In: Proc. of the 17th Conference on Behavioral Representation in Modeling and Simulation.
41. Suchanek, F. M., Kasneci, G., Weikum, G. (2008). Yago: A large ontology from wikipedia and wordnet. In: Journal of Web Semantics, 6(3), 203-217.
42. Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P. N., Bizer, C. (2015). Dbpedia – a large-scale, multilingual knowledge base extracted from wikipedia. In: Semantic web, 6(2), 167-195.
43. Kalender, M., Dang, J., Uskudarli, S. (2010). Unipedia: A unified ontological knowledge platform for semantic content tagging and search. In: 2010 IEEE Fourth International Conference on Semantic Computing, 293-298.
44. Aggarwal, N., Buitelaar, P. (2012,). Query Expansion Using Wikipedia and Dbpedia. In: CLEF (Online Working Notes/Labs/Workshop).
45. Zhou, D., Wu, X., Zhao, W., Lawless, S., Liu, J. (2017). Query expansion with enriched user profiles for personalized search utilizing folksonomy data. In: IEEE Transactions on Knowledge and Data Engineering, 29(7), 1536-1548.
46. Ray, S. K., Singh, S., Joshi, B. P. (2009). Exploring multiple ontologies and WordNet framework to expand query for question answering system. In: Proc. of the First International Conference on Intelligent Human Computer Interaction: (IHCI 2009), 296-305).
47. Deutch, D., Frost, N., & Gilad, A. (2017). Provenance for natural language queries. In: Proc. of the VLDB Endowment, 10(5), 577-588.
48. Unni, M., Baskaran, K. (2011). Overview of approaches to semantic web search. In: International Journal of Computer Science and Communication (IJCSC), 2, 345-349.
49. Sudeepthi, G., Anuradha, G., Babu, M. S. P. (2012). A survey on semantic web search engine. In: International Journal of Computer Science Issues (IJCSI), 9(2), 241-245.
50. Cimiano, P., Haase, P., Heizmann, J., Mantel, M., Studer, R. (2008). Towards portable natural language interfaces to knowledge bases– The case of the ORAKEL system. In: Data & Knowledge Engineering, 65(2), 325-354.
51. Croft, W. B., Turtle, H. R., Lewis, D. D. (1991). The use of phrases and structured queries in information retrieval. In: Proc. of the 14th annual international ACM SIGIR conference on Research and development in information retrieval, 32-45.
52. Teletska, A. O., Zagnitko, A. P., Nadutenko, M. V. (2018). Classification of information search languages. History, philosophy, law, 120. (in Ukrainian)
53. Chowdhury G. G. (2010) Information Retrieval, 3rd edition. London: Facet Publishing, 488 p.

54. Serbin, O. (2008). Representation of information search languages in web-oriented systems. In: Scientific works of the V.I. Vernadskyi National Library of Ukraine, (20), 176-184. (in Ukrainian)
55. Rogushina, J. V. (2019). Means and methods of the unstructured data analysis. In: Problems in programming, (1), 57-77.
56. Pérez, J., Arenas, M., Gutierrez, C. (2009). Semantics and complexity of SPARQL. In: ACM Transactions on Database Systems (TODS), 34(3), 1-45.
57. Weiand, K., Hartl, A., Hausmann, S., Bry, F., Furche, T. (2012). Keyword-Based Search over Semantic Data. Semantic Search over the Web, 159-192.
58. Bao, J., Ding, L., Hendler, J. (2008). Knowledge representation and query in semantic MediaWiki: a formal study. Tetherless World Constellation (RPI) Technical Report. DOI 10.1.1.187.4263.
59. Rogushina, J., Priyma, S., Strokan, O. (2017) Creating and Use of Semantic Wiki Resources: A Study Guide. – Melitopol, 169 p. (in Ukrainian)
60. Rogushina, J., Grishanova, I. (2022) Semantic Information Resources with a Complex Structure: Knowledge Representation, Scaling and Search Problems. In: UkrPROG, CEUR Vol-3501, 158-171.
61. Pryima, S., Rogushina, J., Strokan, O. (2018). Use of semantic technologies in the process of recognizing the outcomes of non-formal and informal learning. In: CEUR Workshop Proceedings, 226-235
62. The FAIR Guiding Principles for scientific data management and stewardship. Available from: https://www.nature.com/articles/sdata2016118.
63. Rogushina, J., Grishanova, I. (2022). Study of principles, models and methods of FAIR paradigm of scientific data management for analysis for BIG data metadata. In: Problems in programming, (4), 26-35.
64. Rogushina, J. (2023). Development of intelligent information analytical webportals based on semantic Wiki technologies: problems and challenges. In: Problems in programming, (3), 66-80.

Одержано: 23.11.2023

Про авторів:

Рогушина Юлія Віталіївна,
Кандидат фіз.-мат.наук, с.н.с.
Інституту програмних систем
НАН України,
публікації в українських виданнях – 200,
публікації в іноземних журналах – 40.
Індекс Хірша: Scopus – 5,
Google Scholar – 20.
ORCIDhttp://orcid.org/0000-0001-7958-2557.

Місце роботи авторів:

Інститут програмних систем
НАН України, 03181, Київ-187,
проспект Академіка Глушкова, 40,
e-mail: ladamandraka2010@gmail.com,
066 550 1999.

УДК 517.958:57 +519.711.3 + 612.51.001 http://doi.org/10.15407/pp2023.04.056

R.D. Grygoryan, A.G. Degoda, T.V. Lyudovyk, O.I.Yurchak

SIMULATING OF HUMAN PHYSIOLOGICAL SUPERSYSTEMS: MODELING OF KIDNEY AND BLADDER FUNCTIONS

A quantitative model describing the functions of human kidney and bladder is created. The model is realized and tested as an autonomous C# software module (SM) functioning under given dynamic input characteristics. Finally, SM will be incorporated into our specialized general software capable of simulating the main modes of human integrative physiology, namely, interactions of physiological super-system (PSS). The model of the kidney describes mechanisms of blood filtration in Bowman's capsule, reabsorption in collecting tubules, as well as the central renin-angiotensin system mechanism. The model of the bladder describes the dynamics of its filling and periodic emptying. Each act of bladder emptying is initiated by a signal generated by the brain in response to afferent impulse patterns from the bladder's mechanoreceptors. Models have been tested using algorithms that design scenarios, including simulation of either short-time or long-time (hours or days) observations. Input data include different combinations of pressure in renal afferent arterioles, osmotic, and oncotic blood pressures. Output data includes dynamics of primary urine, final urine, bladder volume, urine pressure, mechanoreceptors' activity, renin production velocity, blood renin concentration, angiotensin2 production velocity, and blood angiotensin2 concentration, as well as blood albumin and sodium concentrations. Both student-medics and physiologists interested in providing theoretical research can be users of SM.
Keywords: physical health, kidney, bladder, physiological mechanisms, quantitative model, simulator.

Introduction

Human organs and certain anatomical-functional systems (AFS) form very complex functional systems known as physiological super-systems (PSS). The general concept of human PSS [1-3] explained deep cellular mechanisms that determine cells interaction for providing of every AFS's actually optimal parameters. However, traditional empiric physiology possesses not by research technologies capable of establishing the main quantitative laws ruling the functionalities of PSS. Potentially, mathematical models could help in solving this problem. But traditional methodology of modeling is focused on creating models of individual organs functioning under known input disturbances. Such models are not suitable for the theoretical study of integrative physiology. To fill this methodological gap in, we need models that take into account both cell-scale autonomous mechanisms and multi-scale multicellular regulatory mechanisms. Moreover, the methodology itself must contain a new and explicit understanding of the rules for the coexistence of populations (colonies) of specialized cells. Namely, such approach to the modeling and computer simulating of human physiology is proposed in [4-8]. Their main novelty is in combining of multi-level physiological mechanisms for explaining of organism-scale adaptive physiological responses to environmental alterations. In fact, this approach also creates potentials for explaining mechanisms that determine the dynamic multi-parametric shape of human physical health (HPH). Such a theoretical fundament is extreme necessary for the individualization of the medical assessment of HPH.

Special model of the thermoregulatory system (MT) recently was presented in [8]. The main reason for its creation was that MT should be compatible with our other models. For solving our problems in the frame of human PSS, it is sufficient to have an MT containing three body compartments: a core (it represents muscle, subcutaneous tissue, and bone), skin, and blood.

Working versions of autonomous C# software modules (SM) help us during the tuning of main model constants and testing

special regimes of the model functioning autonomously. Such simulations imitate traditional physiological investigations in which the input-output relationships of the isolated organ have been established using animal-based experiments.

The goal of this article is to present our latest development – a complex model describing main functions of kidneys and a bladder.

Mathematical model of kidneys

Different models of kidney function have been proposed to clarify the mechanism and peculiarities of human kidneys functioning under physiological conditions and in certain diseases. Perhaps, within the framework of this article, it is sufficient to cite the following publications [9-15].

1. Physiological background

Kidneys represent a specific organ which has its complex structure and provides multiple functions. Nephrons are structural-functional units of every kidney. A nephron in turn is composed of a Bowman's capsule which surrounds the glomerular capillary loops and participates in the filtration of blood from the glomerular capillaries. Bowman's capsule also creates a urinary space through which filtrate (primary urine) can enter the nephron and pass to the proximal convoluted tubule. The latter provides reabsorption of about 99% of water and solved chemicals like glucose, sodium, potassium, magnesium, chlorine ions. Through convoluted tubule the final concentrated urine enters and accumulates in the bladder, from where it is evacuated as it becomes critically full. So, the main kidney function is in controlling of blood chemical composition and total blood volume. As the latter is one of the main determiners of long-term mean arterial pressure, kidneys play essential role in both circulation and blood chemical composition. The latter is essential for providing due velocities of cell metabolism.

The model of CVS is presented in [6,7], thus we omit its description in detail. Perhaps, it is sufficient to note that our CVS-model currently is the most complex model, including mechanisms of circulation's both acute and long-term control.

The most distinguishing sign of every PSS is that it functionally integrates multiple AFS. The extreme necessity in novel models is conditioned with the practical impossibility to provide quantitative empirical research of complex and multi-level mechanisms governing the interaction of multiple HPSS in physiological or pathological conditions. At the same time, it is clear that often complex pathologies appear because of inadequate functioning one or more functional chains of PSS. So, an alternative – simulative research of human PSS is highly encouraged.

In parallel with the crating of novel models, several models created in the frame of the traditional methodology can be re-vised in the light of the PSS concept. Our current publication, presenting such an approach, illustrates both the modeling technology and the main problems arising under its application to models published earlier [6,7]. These publications were chosen because AFSs of this PSS represent organs and systems that are external providers of cellular metabolism. As to its intracellular optimizers, briefly described in [1] (only the energy aspect), we intend to create a special model that has to include also those nuclear mechanisms that are under cytoplasm-released adaptation factors. Namely, these factors increase/decrease the expression of genes in specific sites of DNA.

Special software does provide both tunings of models and all procedures accompanying a wide range of their simulation research. Such work is too big to be made within a usual short-term research project and described in a usual research article. Therefore, we are publishing each interim developmental result that can also be an autonomous unit-software. The final purpose of this multi-stage developmental project is to provide physiologists-researchers with modern computer-based information technology with dual goals. The first goal is simulation research of human certain complex PSSs. The second goal is to provide professors and medical students with a modern specialized visualization technology capable of helping future medics to better understand the physiological basis of non-trivial pathologies.

2. *A model of blood filtration in glomeruli of a nephron*

Schematically, the functional model of kidneys is shown in fig.1.

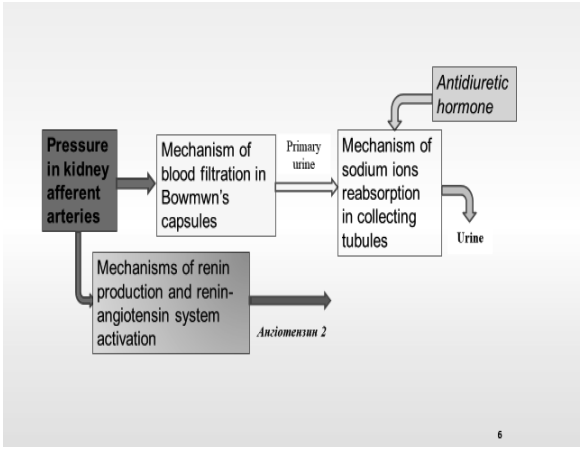

Fig.1. Functional model of kidneys

The equation for the blood filtration dynamics in the glomeruli of the nephron causally relates the velocity of filtration rate $v_k(t)$ to the total (hydrostatic, oncotic, and osmotic) pressure $P_\Sigma(t)$ in Bowman's capsule as:

$$\frac{dv_k(t)}{dt} = \begin{cases} \eta_k \cdot P_\Sigma(t) - v_k(t) & , P_\Sigma(t) > P_{pu}(t) \\ 0 & , P_\Sigma(t) \le P_{pu}(t) \end{cases},$$

$$P_\Sigma(t) = P_h(t) - P_{os}(t) + P_{on}(t),$$

where η_k - is an approximation constant.

A model of sodium reabsorption in the collecting tubules of the nephron is described by a system of differential equations that take into account urine volume (V_u) and sodium concentrations in corresponding sections of tubules ($C_{Na}, C_{Na}^{pc}, C_{Na}^{ct}$):

$$\frac{dNa_{pc}}{dt} = \eta_1 \cdot C_{Na} + Na_{ab,} - Na_{reabpc,} - q_{pc} \cdot C_{Na}^{pc},$$

$$\frac{dV_u}{dt} = q_{ct},$$

$$\frac{dNa_u}{dt} = q_{ct} \cdot C_{Na}^{ct},$$

Hormonal control of the reabsorption dynamics, based on concentrations of albumin (Al) and antidiuretic hormone (Adh), is described by the following system of equations:

$$Na_{reabdc} = S(Al) \cdot Na^*_{reabdc}$$

$$\frac{dAl}{dt} = K_{Al} \cdot \Delta C_{Na}^{KA} - D_{Al} \cdot Al,$$

$$q_{reab}^{ct} = S(Adh) \cdot q_{reab}^{ct*},$$

$$\frac{dAdh}{dt} = K_{Adh} \cdot \Delta C_{Na}^{u} - D_{Adh} \cdot Adh,$$

$$W_{reab} = W^{ct}{}_{reab} + W^{pc}{}_{reab},$$

$$Na_{reabpc} = W^{pc}{}_{reab} \cdot C_{Na}^{pc}.$$

The normalized response of the reabsorption rate to the hormone concentration is described by the function $S(x) = \frac{a \cdot x^n}{b + x^n}$.

3. *A model of the central renin-angiotensin system*

Despite the natural activation of the central renin-angiotensin system (CRAS) mechanism plays only a secondary role in the main function of the kidneys, the fact that the receptor link of CRAS is located precisely in the afferent arterioles of the kidneys deserves our special attention to CRAS. It is one of the powerful hormonal mechanisms of long-term regulation of mean arterial pressure (MAP).

The physiology of CRAS begins with the secretion of renin ($R(t)$) in the kidneys and ends with the formation of the vasoactive agent angitonensin2 ($A^{II}(t)$) in the liver. When modeling, we omit some intermediate transformations. Using the following system of equations, we describe the relationship between the perfusion pressure $P_{aa} - P^T{}_{aa}$ in the kidney afferent arteries and $R(t)$:

$$\frac{dR}{dt} = \begin{cases} \lambda_R \cdot (P_{aa} - P^T{}_{aa}) - R_{ut}, & P_{aa} > P^T{}_{aa} \\ -R_{ut}, & P_{aa} \le P^T{}_{aa} \end{cases}$$

In the second differential equation, which includes the amount of $A^{II}(t)$ in the blood concentration of renin, A_{utI}^{II} is the rate of angitonensin2 molecules of disintegration.

$$\frac{dA^{II}}{dt} = \begin{cases} \delta_{RA} \cdot (R - R^T) - A_{utI}^{II}, & R(t) > R^T \\ -A_{utI}^{II}, & R(t) \le R^T \end{cases}.$$

In these equations, parameters λ_R, $P^T{}_{aa}$, R_{ut}, δ_{RA}, and R^T are approximation constants.

The last three equations take into account the effects of angitonensin2 on vascular parameters in the CVS model:

$$U_i(t) = U_i(0) - \vartheta_i \cdot A^{II},$$

$$D_i(t) = D_i(0) + \vartheta_i \cdot A^{II},$$

$$r_i(t) = Z(V_i(t), D_i(t), U_i(t)),$$

where ϑ_i are approximation constants for each vascular compartment.

4. *A model of bladder's filling-emptying*

A special model for simulations of the periodic dynamics of bladder's filling and emptying is created. In this model, the volume of urine, that gradually accumulates in the bladder and creates pressure in it, is the input variable. This volume is also an input variable of bladder's mechanoreceptors, which have a sensitivity threshold and a saturation level. Therefore, when the afferent pulse rate is approaching to the saturation level, a signal born in the brain is sent in a downward direction to the bladder to create a muscular effort for urine expelling.

For the formal description of these physiological acts following additional variables are introduced: 1) t_b conditional time within the filling-emptying cycle; 2) V_b bladder volume; 3) P_b bladder pressure and volume; 4) D_b stiffness of the bladder; 5) r_b bladder outlet valve resistance; 6) q_b flow of urine; 7) R_{bm} activity of bladder mechanoreceptors; 8) ΔP_b additional pressure.

$$r_b(t_b) = \begin{cases} r_b^{\max}, & P_b(t_b) < P^{\max} \\ r_b^{\max} - (r_b^{\max} - r_b^{\min}) \cdot t_b, & V_b(t_b) \geq V_b^{\min} \end{cases},$$

$$\frac{dV_b}{dt} = q_{ct} - q_b,$$

$$R_{bm}(t) = \begin{cases} 0, & R_{bm}{}^T > P_b(t) \\ \dfrac{1 - e^{(\upsilon \cdot (R_{bm}{}^T - P_b(t))}}{1 + \psi \cdot e^{(\upsilon \cdot (R_{bm}{}^T - P_b(t))}}, & R_{bm}{}^T \leq P_b(t) \leq P_b^S \\ 1, & P_b(t) > P_b^S \end{cases},$$

$$P_b(t_b) = \begin{cases} 0, & V_b(t_b) < V0_b \\ (V_b(t_b) - V0_b) D_b(t_b), & V_b(t_b) \geq V0_b \end{cases},$$

$$q_b(t_b) = P_b(t_b) / r_b(t_b),$$

where $R_{bm}{}^T$, $r_b^{\max}$, $r_b^{\min}$, $P^{\max}$, $V_b^{\min}$, υ, P_b^S, $V0_b$, and ψ - are approximation constants.

Main simulation results

In the frame of this article, we consider test simulations of the autonomous kidney-bladder model. In intact organism, values of input variables are provided by the organs that are directly or indirectly interacting with kidneys.

A simulation of the autonomous kidney-bladder model requires previous setting of both model constants and values of those variables that normally have been provided by other organs. Special window created for providing the tunings for a simulation is shown in fig.2. Once set, the input values are still working for multiple simulations until the user is interested in simulating of effects caused by new input data combination. This article does not provide the reader by simulation data reflecting effects of concrete diseases. All simulations shown below have a goal to demonstrate the fact that our models and the software technology, despite being an interim product yet, are a promising scientific tool. After a proper modification, the proposed software can be used for mining of additional information concerning functions of kidneys and bladder.

Figures 3-6 illustrate dynamics of a group of physiological characteristics during 6-hours simulation in human rest condition in horizontal position.

As shown in fig.3, urine formation depends on differences between primary urine and water reabsorption.

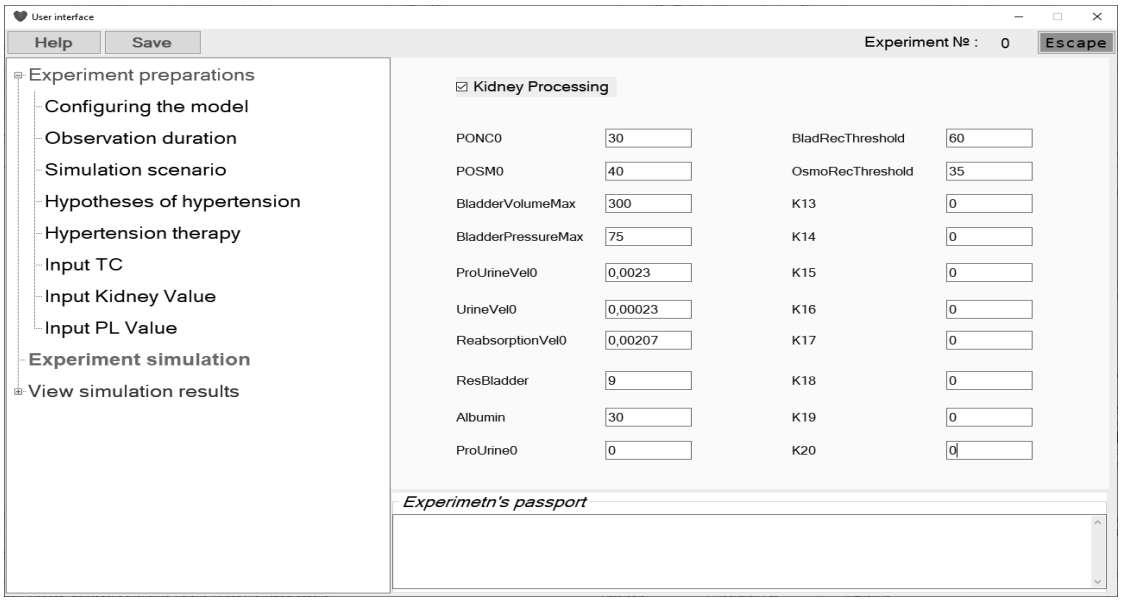

Fig.2. User interface fragment: special window for setting initial data necessary for processing the kidney model.

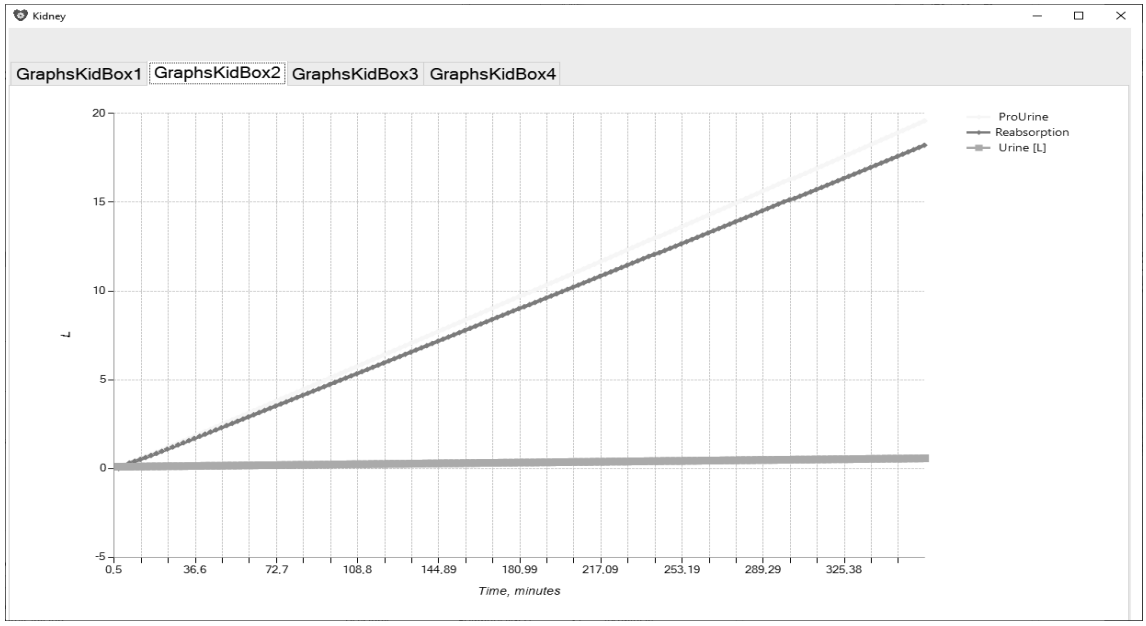

Fig.3. Dynamics of primary urine (Pro-urine) water reabsorption and final urine during six-hours observation (simulation data).

Fig.4. collects dynamics of bladder volume, bladder pressure, blood oncotic, and osmotic pressures. Bladder volume and pressure almost linearly increase with time until bladder mechanoreceptors (their dynamics see in fig. 5) reach a critical level for generating a brain efferent signal (additional pressure to bladder filling pressure) and opening of bladder's output sphincter. Just after this signal, bladder volume and pressure almost linearly decrease to their initial values. This is a simulation of a single cycle of bladder filling-emptying. According to simulation algorithms, such cycles can occur many times if the observation time is bigger than we simulated. Pay attention that in fig.5 the second variable (activity of osmotic receptors) is still at low and practically stable level. Certainly, in case of real and variable composition of blood salts, osmotic receptors will have appropriate activity and the reflex will play a proper role in arterial pressure, blood circulation, and in kidneys function. Fig.6 collects three variables: blood concentrations of albumin, sodium, as well as a conventional variable characterizing the energy status of kidneys. It is well known that the

sodium reabsorption in collecting tubules is an active ion transport against concentration gradients. This is a work provided by use of ATP molecules. Kidneys, despite their relatively little mas, are very "expensive" organ requiring about 20% of ATP synthesized in organism. Namely, this circumstance does play an essential role in efficiency of organ's function under general deficiency of energy sources like carbohydrates and oxygen. These aspects of organism-scale functioning cannot be theoretically analyzed otherwise than using proper models that describe functions of organs depending on their current energy status.

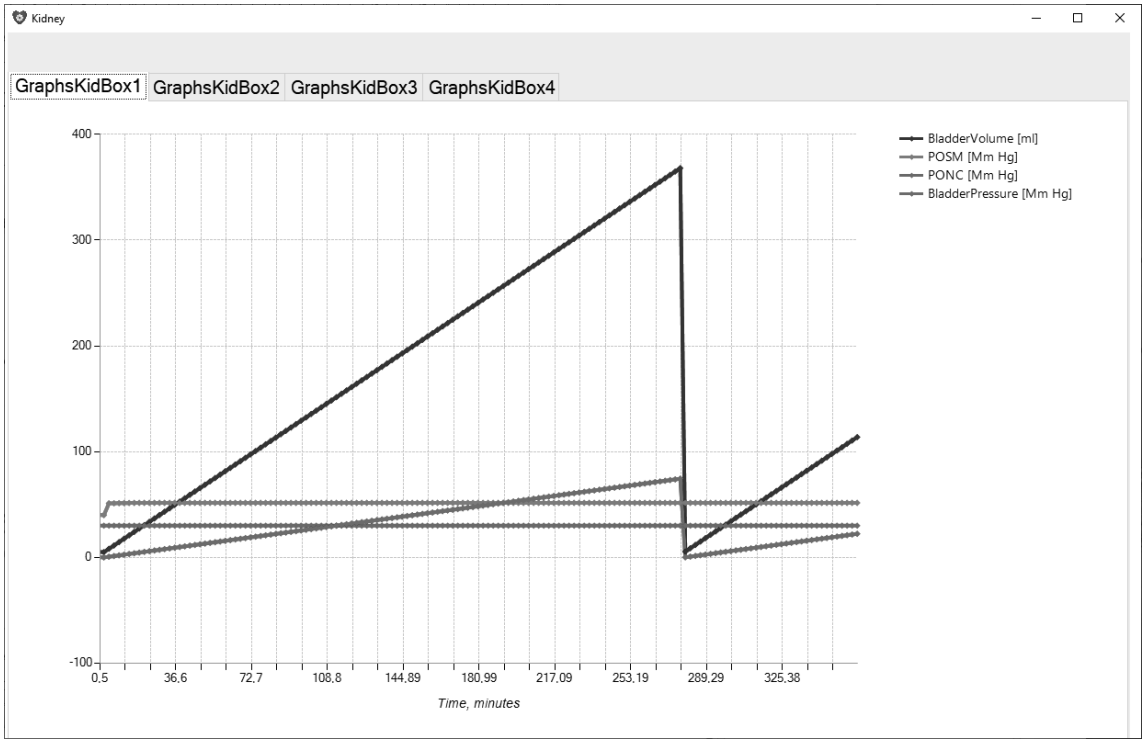

Fig.4. Dynamics of bladder volume and pressure, blood osmotic and oncotic pressures during six-hours observation (simulation data).

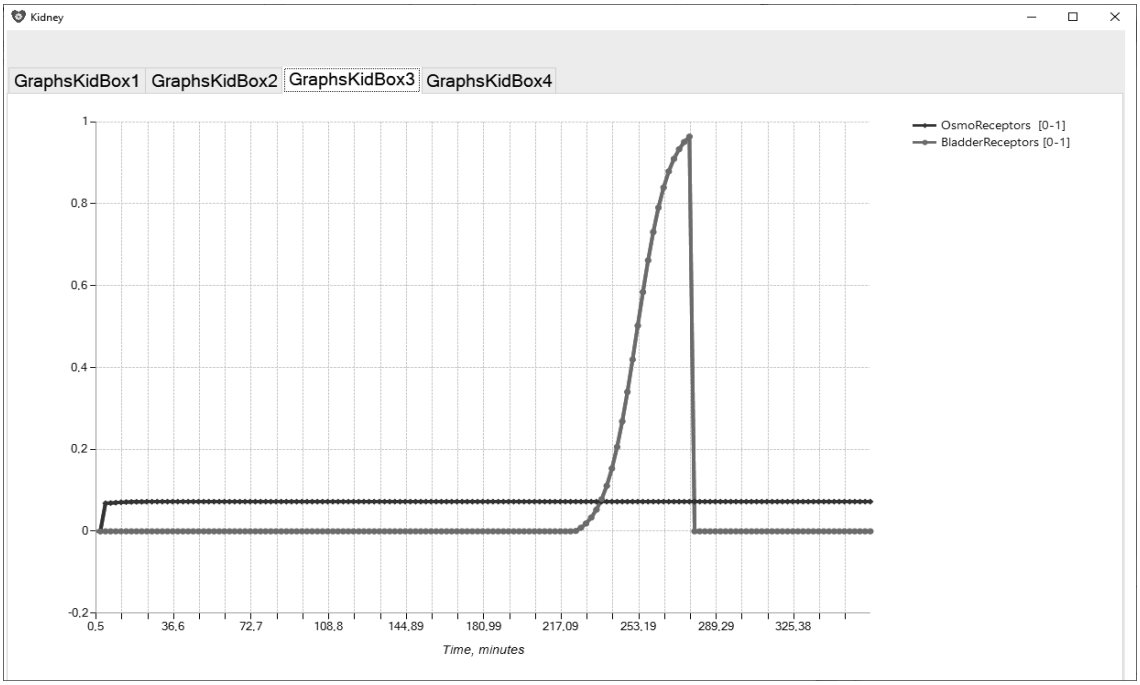

Fig.5. Dynamics of brain osmoreceptors and bladder mechanoreceptors during six-hours observation (simulation data).

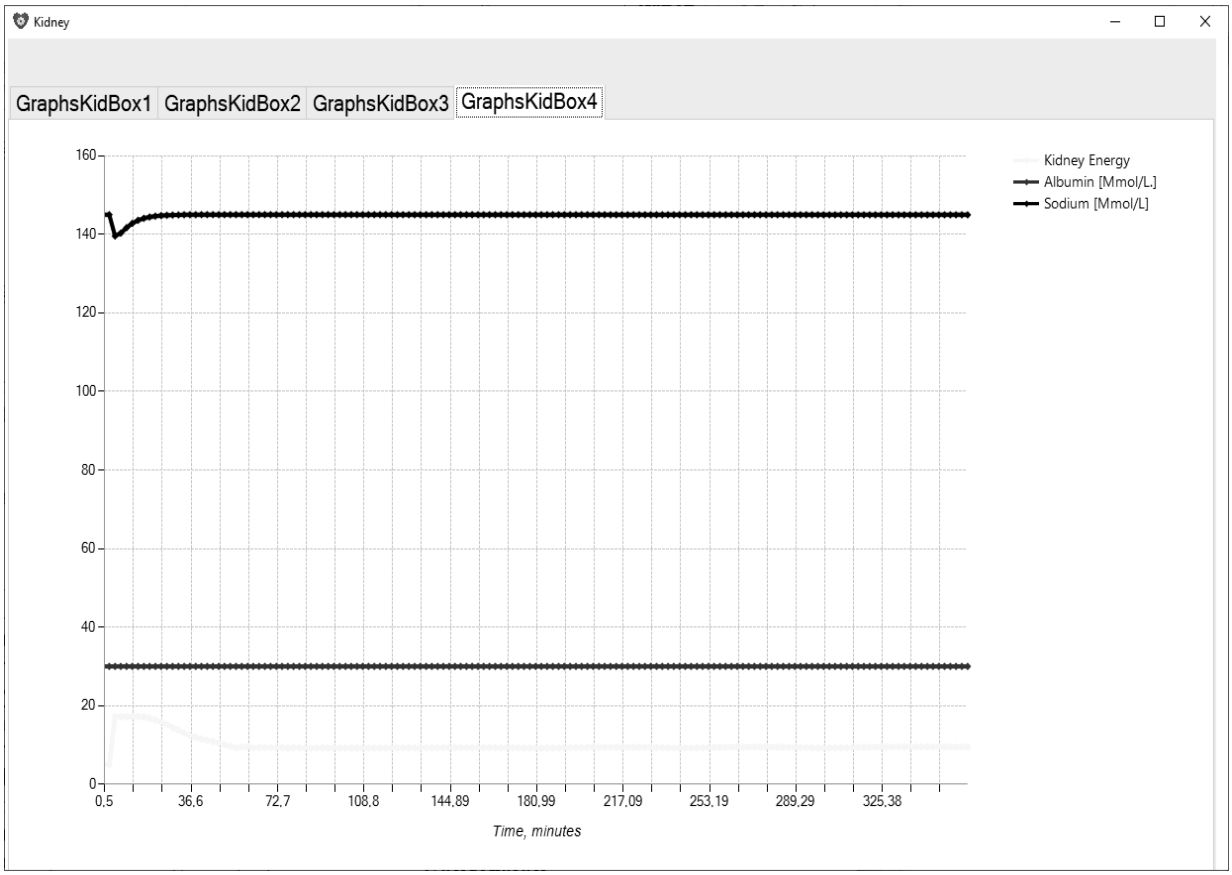

Fig.6. Dynamics of blood albumin, and sodium concentrations, as well as kidneys energy status six-hours observation (simulation data).

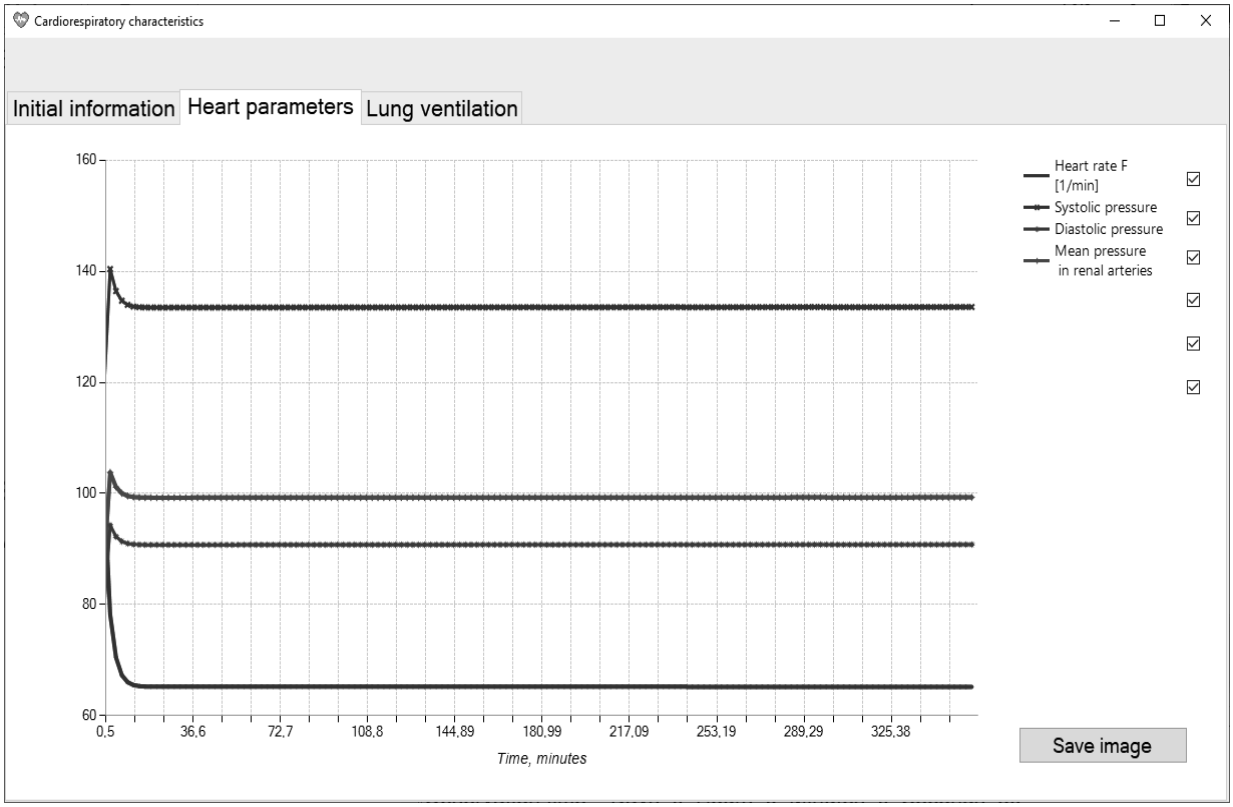

Fig.7. Dynamics of several hemodynamic variables (heart rate, systolic and diastolic pressures, mean pressure in renal arteries) during six-hours observation (simulation data).

Certainly, our simulator yields much more output data concerning not only kidney-bladder functions but also relating to blood circulation parameters, activities of baroreceptors, chemoreceptors, and dynamics of main endocrine hormones. Most hormones modulate not only the state of CVS but also the state of those body structures that concern the functionality of kidneys. An example of additional simulation data contains the fig. 7. It presents dynamics of several hemodynamic variables (heart rate, systolic and diastolic pressures, and the mean pressure in renal arteries) during six-hour observation. We do not present more such additional data for two main reasons. The first one is already mentioned above – the limited paper volume. The second reason is concerned with the "raw" state of the model. Values of several constants and variables are included in the model in conventional units only. We plan to advance the final model when all component models will be created and integrated into the complex simulator of human PSS.

Conclusions

In order to extend the potentials of the PC-based simulator of the human physiological super-system (PSS), a special quantitative model of the human kidneys and a bladder is created and mainly tested. The kidneys model describes: 1) the mechanism of blood filtration in Bowman's capsule and dynamics of primary urine formation; 2) the mechanism of water and sodium reabsorption in conducting tubules; 3) the mechanism of bladder dynamic filling and periodic emptying using afferent patterns of bladder's mechanoreceptors; 4) the central renin-angiotensin-aldosterone (CRAS) mechanism which is one of main determiners of long-term MAP. Algorithms provide designing of scenarios including simulation of either short-time or long-time (hours or days) observations. Test simulations presented in the article covering six-hours observation of kidney-bladder function. Adequateness of models gives us an opportunity to incorporate them into special software-modeling research tool with a final goal to provide theoretical investigations of human PSS.

In a near future, this simulator is to be widened by two more models: 1) of mechanisms controlling lung ventilation; 2) of mechanisms controlling liver-pancreas interaction. These additional mechanisms specifically modulate circulation and cells metabolism.

References

1. Grygoryan R.D., Sagach V.F. The concept of physiological super-systems: New stage of integrative physiology. Int. J. Physiol. and Pathophysiology, 2018: 9,2,169-180.
2. Grygoryan RD. The optimal circulation: cells contribution to arterial pressure. N.Y.: Nova Science,2017: 287p. ISBN 978-1-53612-295-4.
3. Grygoryan RD. The Optimal Coexistence of Cells: How Could Human Cells Create The Integrative Physiology. J. of Human Physiol. 2019,1 (01):8-28. DOI 10.30564/jhp.v1i1.1386.
4. Grygoryan R.D. Problem-oriented computer simulators for solving of theoretical and applied tasks of human physiology. Problems of programming. 2017, №3, 102-111.
5. Grygoryan R.D., Yurchak O.I., Degoda A.G., Lyudovyk T.V. Specialized software for simulating the multiple control and modulations of human hemodynamics. Prombles in programming. 2021; 2: 42-53.
   DOI: https://doi.org/10.15407/pp2021.02.042.
6. Grygoryan R.D., Degoda A.G., Lyudovyk T.V., Yurchak O.I. Simulations of human hemodynamic responses to blood temperature and volume changes. Prombles in programming. 2023; 1: 19-29.
   DOI: https://doi.org/10.15407/pp2023.01.019
8. Grygoryan R.D. Modeling of mechanisms providing the overall control of human circulation. Advances in Human Physiology Research, 2022,4,5 – 21, https://doi.org/10.30564/ahpr.v4i1.4763.
   Grygoryan R.D., Degoda A.G., Lyudovyk T.V., Yurchak O.I. Simulating of human physiological supersystems: interactions of cardiovascular, thermoregu-

latory and respiratory systems. Problems of programming. 2023, №3, P. 81-90. http://doi.org/10.15407/pp2023.03.81.
9. Zaritski R. M. Models of complex dynamics in glomerular filtration rate. Ph.D. dissertation, State University of New York, Buffalo, New York,1999.
10. Thomas S.R. Mathematical models for kidney function focusing on clinical interest. Morphologie, 2019, 103, Issue 343, , 161-168. https://doi.org/10.1016/j.morpho.2019.10.043.
11. Sgouralis I, Layton AT. Mathematical modeling of renal hemodynamics in physiology and pathophysiology. Math Biosci. 2015;264:8-20. doi:10.1016/j.mbs.2015 .02.016.
12. Cupples WA, Braam B. Assessment of renal autoregulation. Am J Physiol Renal Physiol. 2007;292(4):F1105–23.
13. Williamson GA, Loutzenhiser R, Wang X, Griffin K, Bidani AK. Systolic and mean blood pressures and afferent arteriolar myogenic response dynamics: a modeling approach. Am J Physiol Regul Integr Comp Physiol. 2008;295(5):R1502–511.
14. Ryu H, Layton AT. Effect of tubular inhomogeneities on feedback-mediated dynamics of a model of a thick ascending limb. Math Med Biol. 2013;30(3):191–212.
15. Weinstein, A. M. Mathematical models of renal fluid and electrolyte transport: acknowledging our uncertainty. American Journal of Physiology - Renal Physiology. 2003, 284, 871–884.

Received: 10.08.2023

About the authors:

Grygoryan Rafik
Department chief, PhD, D-r in biology
Publications number in Ukraine journals -154
Publications number in English journals -49.
Hirsch index – 11
http://orcid.org/0000-0001-8762-733X.

Degoda Anna,
Senior scientist, PhD.
Publications number in Ukraine journals – 17.
Publications number in English journals -1.
Hirsch index – 3.
http://orcid.org/0000-0001-6364-5568.

Lyudovyk Tetyana,
Senior scientist, PhD.
Publications number in Ukraine journals – 32.
Publications number in English journals -17.
Hirsch index – 5.
https://orcid.org/0000-0003-0209-2001.

Yurchak Oksana,
Leading software engineer.
Publications number in Ukraine journals – 6.
Publications number in English journals - 0.
Hirsch index –0.
https://orcid.org/0000-0003-3941-1555.

***Place of work**:*

Institute of software systems
of Ukraine National Academy of Sciences
03187, Kyïv,
Acad. Glushkov avenue, 40,
Phone.: 526 5169.
E-mail:rgrygoryan@gmail.com,
anna@silverlinecrm.com,
tetyana.lyudovyk@gmail.com,
daravatan@gmail.com

УДК 519.85 http://doi.org/10.15407/pp2023.04.065

Ю.Г. Тарасіч, Г.А. Солошенко

ІНСЕРЦІЙНА СЕМАНТИКА КВАНТОВИХ ВЗАЄМОДІЙ

У статті розглядається застосування методу інсерційного моделювання, а саме – теорії агентів та середовищ на основі алгебри поведінок, до моделювання експериментів, спрямованих на вивчення фізичних, хімічних та біологічних процесів.
Даний підхід використовує багаторівневу модель взаємодій, в основі якої лежать знання про квантові взаємодії в середовищі атомів та молекул. Усі інші види взаємодій (міжмолекулярні взаємодії, хімічні реакції, взаємодії біологічних об'єктів тощо) розглядаються та моделюються як наслідки формалізованих законів квантової взаємодії. Застосування алгебраїчного підходу дозволяє розглядати як експерименти що стосуються вивчення властивостей певного процесу та визначення його кінцевого результату в термінах середовища експерименту, так і можливість оберненого моделювання, коли задано «кінцевий результат» (властивості певної фізичної сутності – речовини або процесу) та необхідно знайти початкові параметри і відповідні дії, що приводять до нього.
У статті наведено приклади формального представлення основних агентів, що розглядаються у моделях, їхні взаємодії, а також дослідження властивостей речовини на прикладі виявлення речовини з магнітоелектричними властивостями.
Ключові слова: інсерційне моделювання, квантові взаємодії, молекулярне моделювання, алгебраїчне моделювання

Вступ

Вивчення квантових взаємодій відіграє надзвичайно важливу роль у створенні нових можливостей для науки, техніки, медицини, комунікації, безпеки та інших сфер діяльності людини.

Квантові взаємодії лежать в основі механізмів виникнення та властивостей сили взаємодії електромагнітної природи між молекулами, що дозволяє розглядати процеси утворення та декомпозиції молекул за різних значень параметрів середовища – температури, тиску, наявності каталізаторів та інших факторів на найнижчому рівні абстракції. Таким чином, вивчення квантових взаємодій розглядається як одна із основних можливостей у відкритті нових речовин із корисними властивостями (суперпровідність, суперпластичність, квантові точки, магнітоелектричність тощо), оскільки дозволяє зрозуміти структуру та властивості речовини на мікроскопічному рівні, а також процеси, які відбуваються в атомах, молекулах, кристалах, наночастинках та інших квантових системах. Сьогодні особливу увагу привертають дослідження, спрямовані на виробництво сильних магнітів [1,2], дослідження поведінки електронів та, зокрема, «квантової заплутаності» [3-5], відкриття нових форм надпровідності [6], отримання чистої енергії [7] тощо. Важливу роль для здійснення досліджень у даних напрямках відіграють методи та системи молекулярного моделювання, зокрема, системи комп'ютерного молекулярного моделювання.

Моделювання взаємодій між мікро та макромолекулами на квантовому рівні дозволяє маніпулювати електронними, магнітними, оптичними та іншими характеристиками речовини, а також розглядати можливості створення нових хімічних зв'язків, молекулярних структур, фазових переходів, квантових станів та ін. Застосування методів квантової механіки для з’ясування закономірностей поведінки мікросистеми дозволяє поширити квантово-механічний підхід на опис фізичних властивостей макросистем, таких як, твердих тіл та рідин, які складаються з великої кількості окремих атомів чи молекул.

Застосування методів моделювання як для неперервних, так і для дискретних моделей, відбувається за допомогою різних підходів: статистичного, ймовірнісного, імітаційного, візуального. Формальні математичні специфікації для опису знань про поведінку атомів, молекул, іонів використовуються в програмних системах, існують стандарти мов формалізованих хімічних та біологічних об’єктів. Наприклад,

формат моделей SBML для обміну та зберігання біологічних моделей, що має широку системну підтримку. Моделювання відбувається як на рівні взаємодії речовин, так і на молекулярному та атомному рівнях. До категорії квантово-хімічних методів обчислень належать такі методи як метод Гартрі-Фока, методи кореляції, теорія функціоналу густини (DFT) та ін.[8,9].

Одним із найпопулярніших методів чисельних квантово-механічних обчислень в обчислювальній фізиці та квантовій хімії є теорія функціоналу густини [10-13]. Проте, не зважаючи на популярність та широке застосовування, метод має певні недоліки, такі як складність представлення міжмолекулярних взаємодій. Зокрема, у врахуванні дисперсійної взаємодії, перенесенні заряду, перехідних станів, тощо [10-12]. Окрім того, хоча обчислювальне навантаження даного методу порівняно з іншими традиційними методами (метод Гартрі-Фока та його сучасні модифікації [13]) значно менше, його вартість з точки зору обчислень залишається досить високою.

Тож, зважаючи на значну складність обчислень та досліджуваних моделей, велику кількість задач в біохімії та біофізиці не може бути ефективно розв'язано за допомогою традиційних методів моделювання через нескінченну кількість можливих сценаріїв поведінки об'єктів. Хоча стохастичне (ймовірнісне) та конкретне імітаційне моделювання ілюструє процеси з точністю до емпіричних даних, повний розгляд множини нескінченних сценаріїв поведінки досліджуваних об'єктів важко здійснити. Одним із потужних інструментів, що набуває сьогодні широкого застосування в біологічних дослідженнях, є нейронні мережі [14-16]. Системи штучного інтелекту (ШІ), побудовані на базі тих чи інших нейронних мереж уже сьогодні дозволяють розв'язувати багато актуальних задач. Так, наприклад, як одна із можливостей підвищення ефективності обчислень традиційних методів моделювання енергії та сил атомних систем, зокрема, методу теорії функціоналу густини, розглядається застосування графових нейронних мереж [17] .

Важливою є можливість імплементації у системи ШІ вже існуючих методів молекулярного моделювання (методів молекулярного докінгу, молекулярної механіки, гібридних квантово-механічних/молекулярно-механічних симуляцій) та глибоких моделей навчання для прогнозування структури, енергії, кінетики та термодинаміки взаємодії молекул. З іншого боку, зважаючи на ряд недоліків використовуваних на сьогодні методів, ми не можемо стверджувати, що отримані нейронною мережею/системою штучного інтелекту результати не є помилковими, хоча дійсно будуть такими, що звужують пошук. Відповідно, постає необхідність у додаткових експериментах та перевірці, підтвердженні отриманих результатів.

Розвиток алгебраїчних систем – машин розв'язування, автоматичного доведення теорем поклав початок новим дослідженням за допомогою символьного моделювання, що дало змогу виводити необхідні знання із множини формалізованих законів. Як бачимо, основна ідея нашого дослідження полягає у застосуванні технології алгебраїчного моделювання та квантово-хімічного апарату для моделювання та перевірки експериментів в галузі фізики, хімії, біології.

Використання алгебраїчної техніки, а саме, – теорії інсерційного моделювання (алгебраїчне моделювання взаємодії агентів та середовищ), створеної видатним академіком О. А. Летичевським (1935-2019), дає змогу здійснювати дослідження на різних рівнях абстракції та оперувати із нескінченними сутностями, що було підтверджено відповідними дослідженнями у галузі розробки надійних систем в електронній промисловості, а також в різних індустріях нашої країни та США.

У даній статті ми розглядаємо можливості та приклади застосування теорії інсерційного моделювання до моделювання квантових взаємодій. У першому розділі статті надано стислий опис застосовуваного методу та інструментів. У другому та третьому розділах наведено приклади формального представлення основних агентів, що розглядаються у моделях (елементарні частинки, атоми, молекули) та їх вза-

ємодії, представлено список деяких досліджуваних властивостей.

Опис підходу

Для моделювання експериментів в царині фізики, хімії та біології з урахуванням необхідності багаторівневого моделювання, в основі якого розглядається моделювання квантових взаємодій, ми пропонуємо алгебраїчний підхід, який реалізується в рамках системи інсерційного моделювання IMS [18], розробленої на базі алгебраїчної системи програмування APS [19].

Інсерційне моделювання зосереджується на побудові моделей та вивченні взаємодії агентів та середовищ у складних багатоагентних системах [20]. Для представлення специфікацій вимог використовується мова алгебраїчних специфікацій дій агентів в алгебрі поведінок. Дія являє собою перехід стану агента з перед- та післяумовою, що представлені логічними формулами в певній теорії. Семантика дій дозволяє створювати конкретні та символьні моделі на різних рівнях абстракції. Таким чином, як математичне уточнення поняття агента ми використовуємо транзиційну систему. Вона є найбільш абстрактною математичною концепцією, моделюючою систему, яка розвивається з часом. У рамках методу інсерційного моделювання квантових взаємодій ми використовуємо специфікації алгебри поведінки для формалізації [18,20]. Як базова логічна мова використовується множина формул логіки першого порядку над поліноміальною арифметикою.

Усі основні концепції, такі як середовище, агенти, базові протоколи, алгебра поведінки тощо, розглядаються у статті на прикладі формалізації будови речовин, молекул, атомів та елементарних частинок (протони, електрони), їхньої взаємодії та зміни їхніх властивостей/характеристик під дією температури, магнітного поля тощо.

Вся семантика фізичних та хімічних взаємодій для моделювання на рівні квантово-хімічного середовища визначається будовою атомів та молекул, а саме будовою ядра, орбіталями, кількістю електронів на відповідних орбіталях та характеристиками електронів. Це забезпечує підтримку базових можливостей, які, в свою чергу, можуть бути використані в моделюванні сутностей більш високого рівня абстракції.

Для моделювання використовуємо систему Insertion Model Creator (Рис.1.), реалізовану на базі системи інсерційного моделювання IMS. За допомогою цієї системи формалізується предметна область та створюється модель задачі, яка буде розв'язуватись у термінах формальних теорій. Далі модель передається до Алгебраїчного серверу, що поєднує в собі формалізовані математичні теорії та відповідні методи, які працюють із моделями та розв'язують задачі [24]. Зокрема, ми маємо своєрідну базу знань, що містить множину формалізованих знань та теорій з квантової фізики, хімії, біології та інших, що забезпечує багаторівневість моделювання.

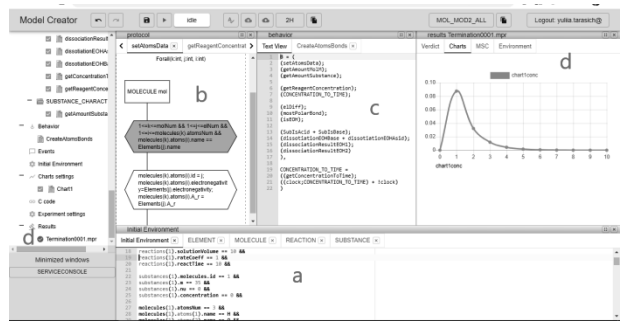

Рис.1. Система Insertion Model Creator: a) Початкова формула середовища, b) Дія, c) Рівняння поведінки, d) Результати моделювання (графіки (для конкретного моделювання), Формула середовища, MSC діаграми,Траса))

Для уникнення можливого явища комбінаторного вибуху через високу складність моделей, що розглядаються, передбачається використання методів нейронних мереж як таких що здатні вказати правильний шлях моделювання. Наприклад, можна визначити можливі варіанти зміни властивостей речовини під дією тих чи інших чинників шляхом оцінки початкового стану середовища, що моделюється. Модель класифікації так само вказує відповідний найефективніший напрямок чи дію, яку треба застосовувати, або відкине напрямки, які ведуть до небажаного результату.

Для моделювання гібридних систем метод алгебраїчного моделювання розширено із можливістю аналітичного розв'язку диференціальних рівнянь, оператори яких є виконуваними алгебраїчними специфікаціями.

Перші результати застосування алгебраїчного підходу до молекулярного моделювання були представлені в [25,26].

Інсерційна семантика міжатомних та міжмолекулярних взаємодій на квантовому рівні

Моделювання взаємодії молекул, органічних та неорганічних речовин на рівні квантово-хімічних механізмів взаємодії потребує визначення властивостей атомів, які входять до їх складу, що говорить про необхідність аналізу їхньої будови та енергетичних станів, що залежать від кількісних та якісних характеристик протонів та електронів як невід'ємних складових кожного атома. Таким чином, для більшості експериментів, протони та електрони розглядаються як складові атома – його основні атрибути. З іншого боку протони та електрони мають власні характеристики та можуть розглядатися як окремі агенти (експерименти з моделювання процесів протонної терапії, роботи прискорювачів частинок, дослідження поведінки електронів тощо). Зважаючи на вищесказане, для формалізації знань найнижчого рівня абстракції – квантових та міжатомних взаємодій, – ми визначаємо такі типи агентів як ELECTRON, PROTON та ATOM.

Як атрибути агента типу ELECTRON розглянуто такі, значення яких відповідають фізичним властивостям електрона та використовуються для моделювання електрон-електронних, міжатомних та міжмолекулярних взаємодій, а саме: маса електрона (атрибут mass), заряд електрона (charge), магнітний момент електрона (orbitalMagneticMoment), імпульс електрона (electronMomentum), швидкість електрона (speed), спін електрона (spinQuantumNum), радіус орбіти електрона (orbitalRadius). Усі перераховані атрибути є атрибутами дійсного типу.

Розгладаючи агенти типу PROTON для більшості експериментів, ми абстрагуємося від будови протона та деяких інших атрибутів та розглядаємо лише такі характеристики як маса та заряд частинки – атрибути mass та charge дійсного типу.

Агентом типу ATOM є частинка, що складається з позитивно зарядженого ядра (до складу якого входять протони та нейтрони) і електронів, які обертаються навколо ядра на різних енергетичних рівнях. Відповідно, використовуючи даний тип агента ми можемо розглядати у моделі як власне самі атоми, так і іони та ізотопи атомів. Основними атрибутами, які визначають даний тип агенту, є:

– атрибути цілочисельного типу: protonsNum, neutronsNum, та electronsNum, які відповідно, визначають кількість протонів, нейтронів та електронів атома/іона/ізотопа; massNumber – масове число, тобто сумарна кількість протонів і нейтронів у ядрі атома (використовується для визначення кількості нейтронів); principalQuantumNum – головне квантове число; orbitalQuantumNum – побічне/орбітальне квантове число; magneticQuantumNum – магнітне квантове число; bondingAbility – кількість зовнішніх/валентних електронів

– атрибути дійсного типу nuclearCharge – заряд ядра; charge – заряд атома/іона; mass – маса атома; ionizationEnergy – енергія іонізації; electronAffinityEnergy – енергія спорідненості до електрону (електронна афінність); electronegativity – електронегативність; radius – радіус атома/іона; effectiveQuantumNum – значення ефективного квантового числа; effectiveCharge – значення ефективного заряду; dipoleMoment – дипольний момент.

– атрибути функціонального типу: electrons(int,int,int,int) – >ELECTRON – зберігає список електронів даного атома. Набуває таких параметрів: номер рівня, номер підрівня, номер орбіталі, номер електрона на орбіталі та повертає агента типу ELECTRON;

– Orbital: (int,int,int) – >int- набуває номер рівня, номер підрівня, номер ор-

біталі і повертає кількість електронів, які знаходяться на даній орбіталі.

Для визначення початкових значень атрибутів агентів типу ATOM, заданих іменем відповідного елемента періодичної таблиці, подальшого визначення атомної будови молекул/речовин та можливості забезпечення багаторівневості моделювання, визначено атрибут name та введено перелічувальний тип Periodic_Elements {H, He, Li, Be, B, C, N, O, F, Ne, Na, Mg,…}, який містить усі назви елементів періодичної таблиці.

Для збереження списку атомів, які розглядаються у моделях, використовується функціональний атрибут atoms: (int) ->ATOM. Даний атрибут може бути як атрибутом середовища, в якому взаємодіють агенти, так і атрибутом агента більш високого рівня, зокрема, молекули. Для визначення кількості атомів, які розглядаються у моделі, використовується цілочислений атрибут atomsNum.

Початкові значення усіх атрибутів середовища (температура, тиск, магнітне поле) та атрибутів агентів задаються початковою формулою середовища та можуть бути задані як конкретно, так і символьно. Наприклад, ми виконуємо конкретне моделювання. Тоді атрибути агентів та середовища матимуть конкретні значення. Для агента типу ATOM, який відповідає хімічному елементу Водень фрагмент початкової формули матиме наступний вигляд:

atoms(1).name== H &&
atoms(1).principalQuantumNum == 1 &&
atoms(1).massNumber == 1 &&
atoms(1).electronegativity == 2.2 &&
atoms(1).Orbital(1,0,1) == 1 &&
atoms(1).electronsspinQuantumNum(1,0,1,1) == 0.5 &&
atoms(1).protonsNum == 1 &&
atoms(1).neutronNum==
atoms(1).massNumber –
atoms(1).protonsNum &&
atoms(1).electronsNum == 1 &&
atoms(1).nuclearCharge ==
atoms(1).protonsNum proton.charge && …*

Досяжність властивості ми також аналізуємо, використовуючи конкретні значення. Виконуючи конкретне моделювання, ми можемо будувати та аналізувати графіки (наприклад, зміна концентрації речовини у процесі реакції залежно від зміни температури тощо), але даний експеримент буде проходити в межах єдиного сценарію.

Будуючи символьну алгебраїчну модель, ми можемо задавати атрибутам агенту довільні початкові значення, наприклад:

atoms(1).name== H &&
atoms(1).principalQuantumNum == 1 &&
1<=atoms(1).massNumber <= 2 &&
atoms(1).electronegativity == 2.2 &&
atoms(1).Orbital(1,0,1) == 1 &&
atoms(1).electronsspinQuantumNum(1,0,1,1) == 0.5 &&
atoms(1).protonsNum == 1 &&
atoms(1).neutronNum==
atoms(1).massNumber –
atoms(1).protonsNum &&
0<=atoms(1).electronsNum <= 1 &&
atoms(1).nuclearCharge ==
atoms(1).protonsNum proton.charge && …*

У даному випадку формула початкового стану експерименту визначає множину можливих сценаріїв, і заданий агент буде розглядатися як атом Водню, або ізотоп Водню, або іон Водню. Такі початкові формули, або алгебраїчні констрейнти можуть бути як завгодно складні. Тоді на кожному кроці алгебраїчного моделювання ми будемо отримувати вже не конкретні числові значення атрибутів, а формулу, що покриває множину сценаріїв. І кінцевим результатом буде не один сценарій, що досягає шуканої властивості, а саме всі сценарії із початкової формули, в яких ця властивість досяжна.

Взаємодії агентів формалізовані за допомогою формальних дій та розглядаються як поведінкові рівняння. Знайти сценарій у вигляді послідовності дій, що веде до шуканої властивості можливо саме через розв’язання поведінкового рівняння, а розв'язок знаходиться за допомогою алгебраїчного моделювання.

Алгебраїчне моделювання є пошуком послідовностей дій в рамках алгебри поведінок. Кожна дія представляє множину переходів, що можливі на кожному кроці моделювання для агента. Результатом буде множина можливих станів, що приводять до певної цілі моделювання.

Як приклад, фрагмент формалізації поведінки моделі, що розглядає утворення міжатомних зв'язків, має наступний вигляд:

```
CREATE_BOND =
(getAtomsBondEnergy; (createIonicBond .
RECALCULATION_ATOMS_DATA +
CreateNonPolarMol .
RECALCULATION_ATOMS_DATA +
CreatePolarMol .
RECALCULATION_ATOMS_DATA +
CreateMetalMol .
RECALCULATION_ATOMS_DATA +
repulsion);
(changeTime.CREATE_BOND +
!changeTime.Delta)),

RECALCULATION_ATOMS_DATA =
(RewriteAtom1Orbitals;
RewriteAtom2Orbitals;
changeAtomsCoordinates; getdistance;
DEFINE_ATOM_CHARACTERISTICS),

DEFINE_ATOM_CHARACTERISTICS = (
getLastLevelElNum;
GET_EFFECTIVE_QUANTUM_NUM;
getAtomMass;
getAtomCharge;
GET_BONDING_ABILITY;
GET_SHIELDING_CONSTANT;
getEffectiveCharge;
getAtomRadius; getTotalElEnergy;
GET_ATOMS_CATIONS_DATA;
GET_ATOMS_ANION_DATA;
getIonizationEnergy;
getElectronAffinityEnergy;
getMallikenElectronegativity)
```

Зазначена поведінка складається з таких дій як getAtomsBondEnergy – визначення енергії, необхідної для утворення зв'язку; createIonicBond – утворення іонного типу зв'язку, CreateNonPolarMol – утворення ковалентного неполярного зв'язку; CreatePolarMol – утворення ковалентного полярного зв'язку; CreateMetalMol – утворення металічного типу зв'язку; repulsion – відштовхування між атомами; changeTime – зміна часу (дозволяє моделювання перебігу експерименту в часі), а також поведінок RECALCULATION_ATOMS_DATA та DEFINE_ATOM_AGENT_CHARACTERISTICS, які складаються з наборів інших дій та поведінок, що дозволяють моделювати зміну значень атрибутів агентів (кількість електронів, які залишились на зовнішньому рівні, маса, радіус, ефективний заряд та ефективне квантове число, енергія іонізації, електронегативність тощо) залежно від типів утворених між ними зв'язків або можливих результатів взаємодії після відштовхування.

Дії визначаються в мові алгебри поведінки, що представляє собою алгебраїчні специфікації. Кожна дія має передумову її спрацювання та зміну стану агента/агентів.

Як приклад розглянемо дію CreateIonicMol, яка визначає утворення іонного зв'язку між атомами:

```
CreateIonicMol = Forall(i:int, j:int, k:int)
((1<=i<=atomsNum && 1<=j<=atomsNum
&& i != j &&
(atoms(i).IonizationEnergy(k) –
atoms(j).ElectronAffinityEnergy(k) +
bondEnergy(i,j)) < 0 &&
atoms(i).electronegativity <
atoms(j).electronegativity &&
atoms(i).bondingAbility !==0 &&
atoms(j).bondingAbility !==0) &&
IFTHE(atoms(i).bondingAbility>=atoms(j).bo
ndingAbility,k==atoms(j).bondingAbility,k==
atoms(i).bondingAbility) &&
(atoms(i).metal==1 && atoms(j).metal==0) )
->
"ATOM#A1: action 'IonicBond';"
(molecules(moleculasNum+1).atoms(molecul
es(moleculasNum+1).atomsNum-1) = i;
molecules(moleculasNum+1).atoms(molecule
s(moleculasNum+1).atomsNum) = j;
molecules(moleculasNum+1).bondEnergy(i,j)
= bondEnergy(i,j);
molecules(moleculasNum+1).bondType(i,j) =
ionic;
molecules(moleculasNum+1).bondV(i,j) = k;
molecules(moleculasNum+1).relativeMolecul
arMass = atoms(i).atomicMass +
atoms(j).atomicMass;
atoms(i).bondingAbility =
atoms(i).bondingAbility – k;
atoms(j).bondingAbility =
atoms(j).bondingAbility – k;
atoms(i).electronsNum =
atoms(i).electronsNum +k ;
atoms(j).electronsNum =
```

atoms(j).electronsNum-k;
moleculasNum=moleculasNum+1),

У передумові ми визначаємо дію для усіх агентів типу ATOM (визначається умовою 1<=i<=atomsNum && 1<=j<=atomsNum та функціоналами atoms(i), atoms(j)), але обираємо два з них, характеристики (значення атрибутів) яких відповідатимуть сукупності наступних умов утворення іонного зв'язку:

– 1) atoms(i).IonizationEnergy(k) – atoms(j).ElectronAffinityEnergy(k) + bondEnergy(i,j)) < 0 – дана умова відповідає визначенню енергетичної вигідності утворення іонної молекули. Тобто даний зв'язок буде утворено за умови, що реакція утворення молекули A^+ B^- є екзотермічною (ΔH=I_A-ε_B+U_0 <0 , де I_A – енергія іонізації атома A, ε_B- енергія спорідненості до електрона атома B, U_0 – кулонівська енергія взаємодії).

– 2) atoms(i).bondingAbility !==0 && atoms(j).bondingAbility !==0 – тут ми перевіряємо які атоми мають вільні електрони, а, отже, – можуть брати участь в утворенні зв'язку (атрибут bondingAbility агента типу атом визначає наявну кількість зовнішніх електронів (валентність)).

3) умова atoms(i).electronegativity < atoms(j).electronegativity визначає, що атоми (i) та (j) мають різну електронегативність, та відповідно дозволяє визначити атом, який забере електрони в результаті утворення зв'язку (i), та атом, який віддасть свої електрони (j).

4) конструкція IFTHE(atoms(i).bondingAbility>=atoms(j).bondingAbility, k == atoms(j).bondingAbility, k ==atoms(i).bondingAbility) визначає кількість електронів (k), які «братимуть участь» в утворенні зв'язку. Тобто, якщо атом, який забирає електрони, потребує більшої кількості електронів (atoms(i).bondingAbility), ніж та, яку може віддати інший атом (atoms(j).bondingAbility), то k = atoms(j).bondingAbility, а в іншому випадку k = atoms(i).bondingAbility.

5) atoms(i).metal == 1 && atoms(j).metal == 0 – перевіряємо, чи один із агентів належить до металів, що визначає, що саме іонний тип зв'язку буде утворено.

Зміна стану визначається постумовою дії та відповідає змінам значень атрибутів агентів, але буде виконана лише у разі, якщо передумова дії виконувана. Для дії CreateIonicMol зміна стану середовища визначає зміну значень атрибутів агентів типу ATOM, які беруть участь в утворенні зв'язку (bondingAbility, electronsNum) та відповідно утворення нового агенту типу MOLECULE (визначається функціоналом molecules(moleculasNum+1)) та визначенням значень деяких її атрибутів (енергія зв'язку – bondEnergy, тип зв'язку – bondType, порядок зв'язку – bondV).

Алгебраїчне моделювання експерименту

Завданням більшості фізичних, хімічних, біологічних експериментів є визначення корисних/необхідних властивостей речовин, або пошук речовини, яка відповідатиме заданим властивостям. За приклад розглянемо експеримент, спрямований на пошук молекулярного магніта або магнітоелектрика, який існуватиме за умови заданої температури.

Для моделювання даного експерименту ми використовуємо знання про електронну будову молекули, формалізовані правила міжатомних та міжмолекулярних взаємодій для визначення характеристики молекули, визначаємо можливість досягнення сценарію за якого суміш молекул/речовин під дією чинників середовища (зміна температури, тиску, тощо) утворить молекулу/речовину із шуканими властивостями.

Усі молекули розглядаються як агенти типу MOLECULE, який з одного боку може розглядатися як агент, що може взаємодіяти з наявними у середовищі атомами та зарядженими частинками, так і виступає агентом вищого рівня, середовищем взаємодії для тих атомів та частинок, з яких складається. Атрибути агента типу MOLECULE представлені числовими значеннями наступних величин: функціонал, що відображає набір атомів, з яких дана молекула складається (atoms), електронна конфігурація молекули (MolOrbital), довжина зв'язку (d_bond),

енергія зв'язку(BondEnergy), дипольний момент (DipoleMoment), молярна маса (M_r), порядок зв'язку (bondMO – за методом молекулярних орбіталей, bondV – за методом валентних зв'язків), тип зв'язку (bondType) тощо.

Оскільки, магнітоелектрик – це матеріал, який за певної температури одночасно є сегнетоелектриком (речовини, які мають спонтанний дипольний електричний момент в одній із кристалічних фаз, що існує в певному діапазоні температур) і парамагнетиком (речовини з невеликою позитивною магнітною сприйнятливістю, які у зовнішньому магнітному полі намагнічуються вздовж поля і дещо підсилюють його). Одночасно йому притаманна магнітострикція (деформація кристалічної структури під дією магнітного поля). Визначення відповідності утворюваного матеріалу вказаним властивостям визначається представленою нижче формулою властивості:

19<=Temperature <= 26 && /*маємо кімнатну температуру*/

molecule(i). isFerroelectrician == true && /*речовина за заданої температури є сегнетоелектриком*/

molecule(i). isParamagnet == true && /*речовина за заданої температури є парамагнетиком*/

molecule(i). isCrystalStructureChanged == true /*речовина змінила свою структуру під дією магнітного поля (змінено конфігурацію електричних диполів і, отже, поляризацію)*/

Алгебраїчне моделювання експерименту відбувається відповідно до поведінкових рівнянь пошуком послідовності дій, що приводять до шуканої властивості. Водночас генерується множина символьних сценаріїв поведінки агентів із використанням довільних значень атрибутів середовища, які можуть бути задані констрейнтами. Досяжність даної формули говорить про досяжність вказаної властивості за певних початкових параметрів середовища (характеристики речовин, з яких буде утворено сплав, початкові значення та коефіцієнти зміни температури, магнітного поля тощо), а саме набуття отриманою речовиною магнітоелектричних властивостей за заданої температури.

Обернене алгебраїчне моделювання дає змогу визначити конкретні початкові дані, за яких досяжність досліджуваної властивості можлива. Обернене моделювання відбувається як моделювання від заданих властивостей до можливої множини початкових атрибутів згідно поведінкового рівняння, що визначають усі можливі взаємодії на даному рівні абстракції. Наприклад, визначення оптимальних показників температури та часу нагрівання/охолодження сплаву, що мають привести до утворення речовини із необхідною електронною будовою тощо.

Представлення сутностей на вищому рівні, що спирається на типи агентів нижчих рівнів, організовується аналогічно. Для речовин, що представлені певним складом молекул, ми розглядаємо допоміжні атрибути. Наприклад, кількість речовини, концентрація. Відповідно до завдання експерименту ми визначаємо атрибути середовища такі, як температура, тиск, наявність та характеристики магнітного поля тощо. Таким чином ми вибудовуємо багаторівневу систему знань, націлену на розв'язання відкритих задач в галузях фізики, хімії, біології та на їх перетині. Дана система знань представляє повний формальний опис моделей, що розглядаються, алгебраїчною мовою та дозволяє виконувати моделювання експериментів на різних рівнях абстракції – від квантових взаємодій до взаємодій між біологічними об'єктами.

Висновки

На сьогодні розроблено досить велику кількість підходів, алгоритмів та засобів побудови моделей, що розглядають міжатомні та міжмолекулярні взаємодії, біофізичні, біохімічні процеси та біологічні системи, однак наявність цілої низки відкритих проблем у галузях органічної та неорганічної хімії, фізики, біології говорить про необхідність удосконалення старих та пошуку нових підходів, інструментів та методів проведення досліджень оригінальних властивостей органічних та неорганічних

речовин. Зокрема, – досліджень із вивчення фізіологічних, біохімічних, фізико-хімічних, молекулярних та квантово-хімічних механізмів їх взаємодії. Окрім необхідності удосконалення та комбінації відомих обчислювальних інструментів, постає необхідність у розробці нових алгоритмів використання квантово-хімічного апарату для аналізу насамперед міжмолекулярних взаємодій, які є базисом для моделювання складних молекул та матеріалів.

Одним із методів, здатним вирішити більшість відкритих питань, є алгебраїчне моделювання, застосування якого, на відміну від інших методів, дає змогу здійснювати дослідження на різних рівнях абстракції та оперувати із нескінченними сутностями. Перевага даного підходу полягає у тому, що знання представляються у алгебраїчному вигляді, зокрема, поведінкових рівнянь. Тож, на відміну від традиційного імітаційного, ймовірнісного моделювання, як, наприклад, у методах докінгу, де передбачається пошук положень молекул для визначення максимальної енергії зв'язності, необхідне розв’язання відповідних рівнянь, що в багатьох випадках дозволяє уникати проблеми експоненційного вибуху. Зважаючи на складність моделей, які розглядаються, додатковим інструментом, що дозволяє уникнути можливого явища комбінаторного вибуху, є застосування методів ШІ та методів обробки Big Data, як таких, що здатні вказати правильний шлях моделювання. Окрім того, застосування алгебраїчного підходу дає можливість виводити наслідки із існуючих законів, а, отже, – може дати нові факти та теорії, що дозволять розв’язати складні завдання. Іншими словами, застосування алгебраїчного підходу у поєднанні з методами нейронних мереж дозволяє визначити та формально довести певні властивості об’єктів (у даному випадку – заряджених частинок, атомів, органічних та неорганічних речовин, клітин, вірусів тощо) процесів, пошук об’єктів чи необхідних значень їхніх параметрів, які відповідають заданим властивостям.

На даному етапі досліджень нами розроблено методологію формалізації складних органічних та неорганічних речовин, хімічних процесів та реакцій, в основі якої лежить формалізація взаємодії атомів та молекул на рівні квантових взаємодій. Моделювання речовин та їх взаємодії на рівні їхньої атомної будови дає механістичне розуміння їхньої поведінки, а використання формальних алгебраїчних методів дозволяє доводити властивості та знаходити релевантні сценарії для ефективного аналізу поведінки різних об’єктів у реальному часі, розглядаючи не окремі сценарії, а множини можливих поведінок.

В основі подальших досліджень – продовження роботи над формалізацією знань квантової фізики та хімії, органічної хімії, біохімії для розширення бази знань системи моделювання. На основі отриманих результатів передбачається продовження розробок формального опису та моделювання процесів міжклітинної та внутрішньоклітинної взаємодії, протонної терапії, досліджень властивостей органічних та неорганічних речовин на базі квантових взаємодій.

Література

1. Метал із космосу може зробити революцію в електроніці – від iphone до винищувачів (2023) ФОКУС. Available at: https://focus.ua/uk/digital/587710-metal-iz-kosmosu-mozhe-zrobiti-revolyuciyu-v-elektronici-vid-iphone-do-vinishuvachiv (Accessed: 21 November 2023).
2. Direct formation of hard-magnetic tetrataenite ... – wiley online library. Available at: https://onlinelibrary.wiley.com/doi/full/10.1002/advs.202204315 (Accessed: 21 November 2023).
3. Di Sante, D. et al. (2022) ‘Deep learning the functional renormalization group’, Physical Review Letters, 129(13). doi:10.1103/physrevlett.129.136402.
4. Krenn, M. et al. (2016) ‘Automated search for new quantum experiments’, Physical Review Letters, 116(9). doi:10.1103/physrevlett.116.090405.
5. Husain, A.A. et al. (2023) ‘Pines’ demon observed as a 3D acoustic plasmon in SR2RUO4’, Nature, 621(7977), pp. 66–70. doi:10.1038/s41586-023-06318-8.
6. Castro, P. et al. (2023) ‘Emergence of the Chern supermetal and pair-density wave through higher-order Van Hove singularities in the haldane-hubbard model’, Physical Re-

view Letters, 131(2). doi:10.1103/physrevlett.131.026601.
7. Вчені розробляють нову нелінійну схему для отримання чистої енергії (2023) Український телекомунікаційний портал. Available at: https://portaltele.com.ua/news/nauka/vcheni-rozroblyayut-novu-nelinijnu-shemu-dlya-otrymannya-chystoyi-energiyi.html (Accessed: 21 November 2023).
8. Atkins, P. and Friedman, R. (2010) 'The foundations of Quantum Mechanics', Molecular Quantum Mechanics [Preprint]. doi:10.1093/hesc/9780199541423.003.0001.
9. Barbosa, N.S., Lima, E.R. and Tavares, F.W. (2017) 'Molecular modeling in Chemical Engineering', Reference Module in Chemistry, Molecular Sciences and Chemical Engineering [Preprint]. doi:10.1016/b978-0-12-409547-2.13915-0.
10. Assadi, M.Hussein. and Hanaor, D.A. (2013) 'Theoretical study on Copper's energetics and magnetism in tio2 polymorphs', Journal of Applied Physics, 113(23). doi:10.1063/1.4811539.
11. van Mourik, T. and Gdanitz, R.J. (2002) 'A critical note on density functional theory studies on rare-gas dimers', The Journal of Chemical Physics, 116(22), pp. 9620–9623. doi:10.1063/1.1476010.
12. Vondrasek, J. et al. (2005) 'Unexpectedly strong energy stabilization inside the hydrophobic core of small protein rubredoxin mediated by aromatic residues: correlated ab initio quantum chemical calculations [J. amer. chem. soc. 2005, 127, 2615−2617].', Journal of the American Chemical Society, 127(22), pp. 8232–8232. doi:10.1021/ja059908l.
13. Johnston, R.L. (2003) 'Book review: Essentials of computational chemistry: Theories and models. by Christopher J. Cramer', ChemPhysChem, 4(4), pp. 402–402. doi:10.1002/cphc.200390072.
14. Sharma, M. and Deswal, S. (2022) 'drugs–protein affinity-score prediction using Deep Convolutional Neural Network', Expert Systems, 39(10). doi:10.1111/exsy.13154.
15. Kuenzi, B.M. et al. (2020) 'Predicting drug response and synergy using a deep learning model of human cancer cells', Cancer Cell, 38(5). doi:10.1016/j.ccell.2020.09.014.
16. Gentile, F. et al. (2022) 'Artificial Intelligence–enabled virtual screening of ultra-large chemical libraries with deep docking', Nature Protocols, 17(3), pp. 672–697. doi:10.1038/s41596-021-00659-2.
17. Zitnick, L., et al. (2022). 'Spherical channels for modeling atomic interactions' Advances in Neural Information Processing Systems, 35, pp. 8054-8067.
18. Letichevsky, A., Letychevskyi, O. and Peschanenko, V. (2016) 'Insertion Modeling and Its Applications', Computer Science Journal of Moldova, 24 (3), Pp. 357-370.
19. APS and IMS are best for rewriting and modelling (2023). Available at: http//www.apsystem.org.ua (Accessed: 21 November 2023).
20. Letichevsky, A. and Gilbert, D. A. (1999) 'Model for Interaction of Agents and Environments', Recent Trends in Algebraic Development Techniques, 1827, pp.311-328.
21. Baranov, S. et al. (2003) 'Leveraging UML to deliver correct telecom applications', UML for Real, pp. 323–342. doi:10.1007/0-306-48738-1_15.
22. .Letichevsky, A.A. et al. (2005) 'System Specification by Basic Protocols',. Cybernetics and System Analyses, 41, pp. 479–493.
23. Letichevsky, A. et al. (2005) 'Basic protocols, message sequence charts, and the verification of requirements specifications', Computer Networks, 49(5), pp. 661-675.
24. Letychevskyi, O., Peschanenko, V. and Volkov, V. (2022) 'Algebraic virtual machine and its applications', Information and Communication Technologies in Education, Research, and Industrial Applications, pp. 23–41. doi:10.1007/978-3-031-20834-8_2.
25. Letychevskyi, O. et al. (2022) 'Algebraic modeling of molecular interactions', Communications in Computer and Information Science, pp. 379–387. doi:10.1007/978-3-031-14841-5_25.
26. Letychevskyi, O. et al. (2023) 'Algebraic Modeling System for Supporting Research in Medicine and Pharmacology', Proceedings of the The12th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, 2, pp. 1093-1098

Одержано: 23.09.2023

Про авторів:

Тарасіч Юлія Геннадіївна,
докторантка.
Кількість публікацій в українських виданнях – 16.
Кількість зарубіжних публікацій – 21.
Індекс Хірша – 3 (Scopus).
https://orcid.org/0000-0002-6201-4569.

Солошенко Ганна Олександрівна,
учителька математики та фізики,
аспірантка
Кількість публікацій в українських виданнях – 5.
Кількість зарубіжних публікацій – 2.
https://orcid.org/0000-0001-9622-310X

Місце роботи авторів:

Інститут кібернетики імені В. М. Глушкова НАН України,
03187, м. Київ,
проспект Академіка Глушкова, 40.
Тел.: (098) 002 8534.
E-mail: yutarasich@gmail.com,

Херсонський науковий ліцей Херсонської обласної ради
73000, м. Херсон,
вулиця Полтавська, 89.
Тел.: +380951038682
E-mail: hannasoloshenko@gmail.com

УДК 004.8 http://doi.org/10.15407/pp2023.04.076

Я. В. Омельяненко

МОДЕЛЮВАННЯ АВТОНОМНОГО ПРОХОДЖЕННЯ ЛАБІРИНТУ З ВИКОРИСТАННЯМ АЛГОРИТМУ NEAT

Автономне проходження лабіринту є класичним завданням прикладної математики, що стосується галузі автономної навігації. У цій роботі ми розглянемо, як можна використовувати методи нейроеволюції для вирішення задач проходження лабіринту. Також тут пояснимо, як визначити функцію пристосованості з використанням оцінки пристосованості агента-навігатора, розрахованої як похідної від відстані агента до кінцевої мети. Ми деталізуємо основи навчання автономного навігаційного агента з використанням методів нейроеволюції і на їх базі далі зможемо створювати більш розвинутий вирішувач для лабіринтів. Під час роботи здійснено розробку симулятора автономного робота, керованого штучною нейронною мережею, продукованою за допомогою алгоритму NEAT. Також розроблені нові методи візуалізації, які полегшують розуміння результатів виконання алгоритму. Всі розробки здійсненні з використанням мови програмування Python.

Ключові слова: генетичні алгоритми, нейроеволюція наростаючих топологій, автономне проходження лабіринту, NEAT, симулятор автономного робота.

Вступ

Завдання проходження лабіринту є класичною проблемою прикладної математики, яка тісно пов'язана зі створенням автономних агентів навігації, здатних знайти шлях у неоднозначних середовищах [1]. Навколишнє середовище у вигляді лабіринту – класична ілюстрація цілого класу проблем, які мають оманливий ландшафт пристосованості [2, 3, 8]. Це означає, що цілеорієнтована функція пристосованості (goal-oriented fitness function) може мати круті градієнти показників пристосованості в глухих кутах лабіринту, близьких до кінцевої мети. Такі зони лабіринту стають локальними оптимумами для алгоритмів пошуку на основі близькості до мети, які можуть сходитися в цих зонах. Коли алгоритм пошуку застряє у такому оманливому локальному оптимумі, він не може знайти адекватного агента, здатного пройти лабіринт.

На рис. 1 зображено двомірний лабіринт, в якому затемнені локально оптимальні глухі кути.

Конфігурація лабіринту на цьому рисунку демонструє ландшафт оманливих показників пристосованості, зосереджених у локально оптимальних глухих кутах (помічених як зафарбовані сегменти). Агент-вирішувач лабіринту, що переміщається від початкової точки (нижнє коло) до точки виходу (верхнє коло) та навчений на основі критерію близькості до мети, буде схильний застрягати в локальних глухих кутах. Крім того, подібна оманлива оцінка пристосованості може завадити алгоритму навчання на основі близькості до мети знайти успішний вирішувач лабіринтів.

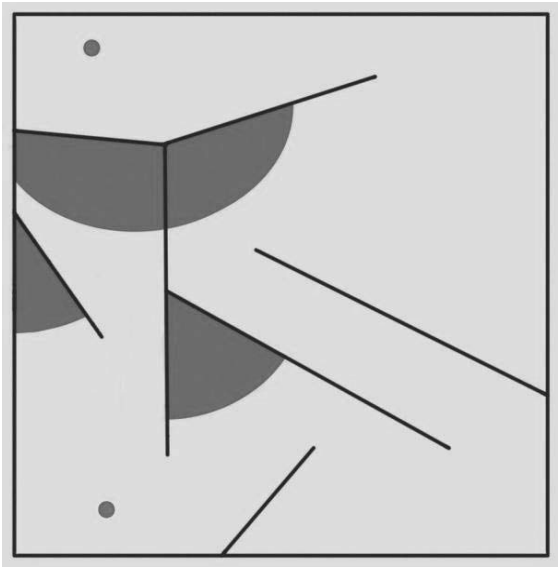

Рис. 1. Двовимірний лабіринт із локально оптимальними глухими кутами (затемнені)

У цій статті ми розглянемо, як алгоритм нейроеволюції NEAT [4, 7, 9] може бути використаний для навчання контролюючої штучної нейронної мережі (ШНМ), яка дасть змогу агенту-вирішувачу пройти лабіринт. Крім того,

будуть розглянуті нові методи, розроблені для візуалізації роботи алгоритму, що дозволяють наочно оцінювати якість його роботи. Розглянуті методи навчання контролюючої ШНМ можуть бути використані для широкого кола задач, пов'язаних із навчанням роботизованих агентів.

1. Огляд аналогів

Як було зазначено вище, задача проходження лабіринту є класичною проблемою прикладної математики, яка використовується для бенчмаркінгу навігаційних алгоритмів.

Для вирішення подібних задач у [5] автори пропонують використовувати еволюційні алгоритми, що фокусуються на навчанні агентів-вирішувачів за принципом максимальної розбіжності знайдених рішень. Але водночас поштовх до розбіжності рішень балансується цілеорієнтованою функцією для звуження простору пошуку. Як результат, продукується процес навчання під назвою *різноманітність якості* (quality diversity, QD). Відомими представниками цього класу алгоритмів є *пошук новизни з локальною конкуренцією* (novelty search with local competition, NSLC) та *багатовимірний архів фенотипових еліт* (Multi-dimensional Archive of Phenotypic Elites, MAPE).

Інший підхід для вирішення задачі проходження лабіринту представлений у [6], де автори пропонують використовувати принципи необмеженої еволюції, яка базується на коеволюції популяцій агентів-вирішувачів та популяцій ШНМ продукуючих конфігурації лабіринтів. Ця ідея базується на засадах збереження *мінімального критерію коеволюції* (minimal criterion coevolution, MCC). Разом з тим мінімальними критеріями, обмежуючими репродуктивну здатність організмів у популяції, є наступне: кожен лабіринт має бути розв'язаний принаймні одним агентом-вирішувачем та кожен агент-вирішувач має розв'язати принаймні один лабіринт. Кожен організм з обох родоводів, що відповідає цьому обмеженню, може розмножуватись. Автори показали, шо цей метод може бути ефективно використаний для пошуку універсальних агентів-вирішувачів.

Недоліками розглянутих підходів є необхідність у значних обчислювальних потужностях, а також тривалий час пошуку рішень.

Стаття пропонує алгоритм, який дозволяє навчати агента, що вирішує задачу лабіринту, за допомогою алгоритму NEAT та мови програмування Python. Розроблений алгоритм навчання може працювати на звичайному комп'ютері та швидко знаходити рішення.

У наступному розділі ми розглянемо основні особливості алгоритму NEAT.

2. Особливості алгоритму NEAT

Найважливішою особливістю алгоритму NEAT, яка визначає його потенціал, є здатність розвивати топологію ШНМ під час процесу навчання.

Важливість поступового нарощування складності топології ШНМ стає очевидною, якщо розглянути, як це реалізовано в алгоритмі NEAT.

Досліджуючи простір пошуку рішення, алгоритм NEAT починає з найпростішої базової топології мережі, яка включає лише вхідні та вихідні вузли ШНМ розв'язувача. Після цього з кожною епохою еволюції топологія вирішувачів ускладнюється, та найскладніші багатовимірні топології оцінюються лише на завершальних етапах еволюційного процесу. До того ж, найбільш підходящі топологічні структури, знайдені під час еволюції, зберігаються алгоритмом NEAT через усі покоління популяції та є предметом наступних оцінок придатності в майбутніх поколіннях. Тобто в процесі еволюції виживуть лише корисні топологічні структури, складність яких завжди виправдана цільовою функцією.

Як бачимо, за рахунок *наростаючого ускладнення топології ШНМ* досягається значне підвищення продуктивності алгоритму NEAT порівнянно з іншими еволюційними алгоритмами. Це досягається за рахунок значного звуження простору пошуку рішення під час навчання з одночасним збереженням різноманітності популяції рішень.

Різноманітність популяції рішень досягається за рахунок іншої суттєвої осо-

бливості алгоритму – *видоутворення* (speciation). Видоутворення дозволяє зберігати корисні мутації за рахунок обмеження репродуктивної конкуренції рамками окремої видової ніші у популяції. Розподіл організмів на види відбувається відповідно до генетичної відстані, тобто подібності між геномами організмів. Як результат, у процесі видоутворення захищаються потенційно корисні інновації та зберігається різноманітність популяції. Це відбувається за рахунок запобігання ситуації, коли найефективніші на даний час рішення витісняють всі інші рішення в межах популяції в цілому. Іншими словами, видоутворення зменшує негативний еволюційний тиск на молоді організми, які були щойно утворенні, та мають більш складну топологію за рахунок додавання нового вузла чи зв'язку між існуючими вузлами. Подібне ускладнення топології може призвести до тимчасового зниження придатності молодого організму. Але водночас подібна інновація може ввести новий вектор для дослідження простору пошуку рішень, що зрештою приведе до винайдення ефективного остаточного рішення у майбутніх поколіннях.

Пошук інновацій разом із запобіганням зупинці у локальних мінімумах цільової функції досягається за рахунок застосування *оператора мутації* до хромосом організмів під час репродукції. Цей оператор змінює один чи декілька генів у геномі організму відповідно до ймовірності мутації, що задається експериментатором як гіперпараметер. За рахунок випадкових змін, що вносяться у геном розв'язувача, досягається збереження генетичної різноманітності популяції та розширення простору пошуку можливих рішень. Оператор мутації є спільною рисою майже всіх генетичних алгоритмів. Але відмінною рисою реалізації його алгоритмом NEAT є те, що змінюються не лише вагові коефіцієнти зв'язків між вузлами ШНМ, а й сама структура топології ШНМ.

Для збереження інновацій та оптимізації обчислень упродовж кросинговеру між організмами популяції, алгоритм NEAT вводить поняття *історичних позначок* (innovation number). Це є однією з найвидатніших особливостей алгоритму NEAT, яка дозволяє оптимально вирішувати проблему пошуку перетину між геномами зі схожими топологіями без потреби в складних обчисленнях на основі графів.

В основу еволюційного процесу під керуванням алгоритму NEAT покладено процес *кросинговеру* (рекомбінації) геномів під час репродукції наприкінці кожної епохи еволюції. Саме на цьому етапі набувають значення історичні позначки, що дозволяють ефективно ідентифікувати гени, які перекриваються (overlapping/matching), а також роз'єднані (disjoint) та надлишкові (excess) гени. Гени, що перекриваються, мають однакові історичні позначки в геномах обох батьків і кодують однакову топологічну структуру незалежно від значень інших параметрів (коефіцієнт ваги з'єднання тощо). В процесі кросинговеру гени, що перекриваються, вибираються випадково у кожного із батьків для наслідування нащадками. Разом з тим, гени, які представлені лише у геномі одного із батьків розрізнюються за положенням їхніх історичних позначок у геномі іншого батька. Якщо історична позначка гена знаходиться в межах діапазону історичних позначок іншого з батьків, то він буде належати до роз'єднаних (disjoint), а якщо виходить за межі - надлишкових (excess) генів. Під час кросинговеру роз'єднані та/або надлишкові гени вибираються випадково від найефективнішого із батьків, або випадково від кожного із батьків (залежно від типу кросинговеру).

Згадані особливості алгоритму NEAT роблять його одним із найбільш вживаних та потужних з-поміж сімейства еволюційних алгоритмів, що дозволяють використовувати його для розв'язання широкого кола прикладних задач. Далі ми перейдемо до розгляду застосування алгоритму NEAT для пошуку ефективного рішення проблеми навігації у лабіринті.

3. Середовище моделювання задачі лабіринту

У рамках цієї статті було розроблено *новітнє програмне забезпечення* для моделювання задачі проходження лабірин-

ту мовою програмування *Python*. Воно складається з трьох основних компонентів, які реалізовані у вигляді окремих класів:

– *Agent* – клас, який зберігає інформацію, пов'язану з агентом навігатора лабіринту, задіяного в симуляції.

– *AgentRecordStore* - клас, який управляє зберіганням записів, що належать до оцінок усіх вирішальних агентів під час еволюційного процесу. Зібрані записи можна використовувати для аналізу еволюційного процесу після його завершення.

– *MazeEnvironment* – клас, який містить інформацію про середовище моделювання лабіринту. Цей клас також надає методи, які керують середовищем моделювання, керують положенням вирішального агента, виявляють зіткнення зі стінами лабіринту та генерують вхідні дані для датчиків агента.

Далі ми розглянемо кожний компонент середовища моделювання лабіринту докладніше.

Агент-вирішувач лабіринту. Агент, що переміщується лабіринтом, являє собою симуляцію робота, обладнаного набором датчиків, що дозволяють йому виявляти прилеглі перешкоди і визначати напрямок виходу з лабіринту. Переміщення робота здійснюється двома приводами, що впливають на лінійний та кутовий рух корпусу робота. Приводи робота управляються нейромережею, яка отримує дані від датчиків і видає два управляючі сигнали на приводи.

Наразі ми розглянемо завдання проходження двовимірного лабіринту. Це завдання легко візуалізувати, і відносно легко написати симулятор робота-навігатора для двомірного лабіринту. Основна мета робота – прохід лабіринтом до певної мети за вказану кількість кроків симуляції. Роботом управляє нейромережа, яка розвивається у процесі нейроеволюції.

Алгоритм нейроеволюції починається з дуже простої початкової конфігурації нейромережі, яка має тільки вхідні вузли для датчиків і вихідні вузли для приводів і поступово стає все складнішим, поки не буде знайдено успішного вирішувача лабіринтів. Це завдання ускладняється особливою конфігурацією лабіринту, в якій є кілька глухих кутів, що заважають знайти шлях до мети і заманюють агента в локальні оптимуми ландшафту пристосованості, як згадувалося раніше.

На рис. 2 представлено схематичне зображення агента-вирішувача, виконаного у вигляді робота та задіяного у моделюванні вирішення задачі лабіринту. На цій схемі зафарбоване коло позначає твердий корпус робота. Стрілка всередині зафарбованого кола показує напрямок руху робота. Шість стрілок навколо зафарбованого кола представляють шість далекомірних датчиків, які вказують відстань до найближчої перешкоди у заданому напрямку. Чотири сегменти зовнішнього кола позначають чотири радарних датчика із секторним оглядом, які діють як компас, що вказує напрямок до мети (виходу з лабіринту).

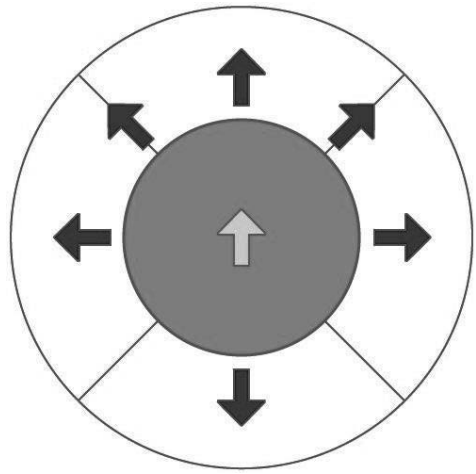

Рис. 2. Схематичне зображення навігаційного агента

Спеціальний радарний датчик активується, коли лінія від точки цілі до центру робота потрапляє у його поле зору (field of view, FOV). Дальність виявлення датчика обмежена зоною лабіринту, яка потрапляє в його поле зору. Таким чином, у будь-який момент часу активований один з чотирьох радарних датчиків, що вказує напрямок виходу з лабіринту.

Радарні датчики мають такі зони огляду щодо курсу робота:

Таблиця 1

Датчик	Поле огляду, градуси
Передній	315,0 ~ 405,0
Лівий	45,0 ~ 135,0
Задний	135,0 ~ 225,0
Правий	225.0 ~ 315.0

Далекомірний датчик – це стежачий промінь, спрямований від центру робота у певному напрямі. Активується під час перетину з будь-якою перешкодою та повертає відстань до виявленої перешкоди. Дальність виявлення цього датчика визначається конкретним параметром конфігурації.

Далекомірні датчики робота відстежують наступні напрямки щодо напрямку руху:

Таблиця 2

Датчик	Напрям, градуси
Правий	-90,0
Передній правий	-45,0
Передній	0,0
Передній лівий	45,0
Лівий	90,0
Задній	-180,0

Рух робота контролюється двома приводами, що прикладають сили, які повертають та/або рухають корпус робота, тобто змінюють його лінійну та/або кутову швидкість.

У реалізації робота-навігатора мовою Python є кілька полів для зберігання його поточного стану та для підтримки станів активності його датчиків:

```
def __init__(self, location, heading=0,
speed=0, angular_vel=0, radius=8.0,
range_finder_range=100.0):
        self.heading = heading
        self.speed = speed
        self.angular_vel = angular_vel
        self.radius = radius
        self.range_finder_range =
range_finder_range
        self.location = location
        # Визначаємо датчики дальноміру
        self.range_finder_angles = [-90.0, -
45.0, 0.0, 45.0, 90.0, -180.0]
        # Визначаємо радарні датчики
        self.radar_angles = [(315.0, 405.0),
(45.0, 135.0), (135.0, 225.0), (225.0, 315.0)]
        # Список станів активності
дальномірів
        self.range_finders = [None] *
len(self.range_finder_angles)
        # Список станів активності
секторних радарів
        self.radar = [None] *
len(self.radar_angles)
```

Цей блок коду демонструє, як створити клас *Agent* зі стандартним конструктором, який задає всі атрибути агенту. Ці атрибути будуть використовуватися симулятором середовища лабіринту для відстеження поточного положення агента на кожній ітерації симуляції.

Реалізація середовища моделювання лабіринту. Щоб змоделювати поведінку агента-вирішувача, що досліджує лабіринт, нам потрібно визначити середовище, яке керує конфігурацією лабіринту, відстежує положення агента-вирішувача і забезпечує вхідні дані для даних датчика навігаційного робота.

Всі ці функції поміщаються в один логічний блок, інкапсульований в клас *MazeEnvironment*, що має наступні поля:

```
def __init__(self, agent, walls, exit_point,
exit_range=5.0):
        self.walls = walls
        self.exit_point = exit_point
        self.exit_range = exit_range
        # Агент-вирішувач лабіринта
        self.agent = agent
        # Прапор індикації про те, що вихід
        знайдено
        self.exit_found = False
        # Початкова відстань від агента до
        виходу
```

```
self.initial_distance =
self.agent_distance_to_exit()
```

У попередньому коді показано конструктор класу *MazeEnvironment* з ініціалізацією всіх його полів:

– конфігурація лабіринту визначається списком стін і точки виходу. Стіни (*walls*) – це списки відрізків; кожен відрізок лінії представляє певну стіну в лабіринті, а точка виходу (*exit_point*) - місце розташування виходу з лабіринту;

– у полі *exit_range* зберігається значення відстані до точки виходу, що визначає зону виходу. Ми вважаємо, що агент успішно пройшов лабіринт, коли його позиція знаходиться у зоні виходу;

– поле *agent* містить посилання на ініціалізований клас агента, описаний раніше, який визначає початкове місце розташування агента в лабіринті та інші поля агента;

– поле *initial_distance* зберігає відстань від початкової позиції агента до точки виходу із лабіринту. Це значення буде пізніше використано для розрахунку показника пристосованості агента.

Генерація сенсорних сигналів. Агент, що проходить лабіринт, управляється нейромережею, якій необхідно мати дані датчиків на вході для формування відповідних керуючих сигналів на виході. Як ми вже згадували, робот-навігатор оснащений масивом двох типів датчиків:

– шість далекомірних датчиків для запобігання зіткненням зі стінами лабіринту, які показують відстань до найближчої перешкоди у певному напрямку;

– чотири секторні радарні датчики, які вказують напрямок до точки виходу з лабіринту з будь-якого його місця.

Показники датчиків необхідно оновлювати на кожному кроці моделювання, а клас *MazeEnvironment* надає два посилальних методи, які оновлюють датчики обох типів.

Масив далекомірних датчиків оновлюється в такий спосіб:

```
for i, angle in enumera-
te(self.agent.range_finder_angles):
        rad = geometry.deg_to_rad(angle)
        projection_point = geometry.Point(
                x = self.agent.location.x +
math.cos(rad) *
self.agent.range_finder_range,
                y = self.agent.location.y +
math.sin(rad) * self.agent.range_finder_range
        )
        projec-
tion_point.rotate(self.agent.heading,
self.agent.location)
        projection_line = geometry.Line(a =
self.agent.location,  b= projection_point)
        min_range =
self.agent.range_finder_range
        for wall in self.walls:
                found, intersection =
wall.intersection(projection_line)
                if found:
                        found_range = intersec-
tion.distance(self.agent.location)
                        if found_range <
min_range:
                                min_range =
found_range
        # Зберігаємо відстань до
найближчої перешкоди
        self.agent.range_finders[i] =
min_range
```

Цей код перераховує всі напрямки далекомірних датчиків, які визначаються кутами напрямку. Потім для кожного напряму вибудовується лінія проекції, починаючи з поточної позиції робота і з довжиною, що дорівнює дальності виявлення далекоміра. Після цього лінія перевіряється на предмет її перетину зі стінами лабіринту. Якщо виявлено кілька перетинів, відстань до найближчої стіни зберігається як поточне значення для конкретного далекоміра. В іншому випадку як поточне значення буде збережена величина максимальної дальності виявлення.

Масив секторних радарних датчиків оновлюється за допомогою коду в класі *MazeEnvironment*:

```
def update_radars(self):
        target = geometry.Point(self.exit_point.x, self.exit_point.y)
        target.rotate(self.agent.heading, self.agent.location)
        target.x -= self.agent.location.x
        target.y -= self.agent.location.y
        angle = target.angle()
        for i, r_angles in enumerate(self.agent.radar_angles):
                self.agent.radar[i] = 0.0
                if (angle >= r_angles[0] and angle < r_angles[1]) or (angle + 360 >= r_angles[0] and angle + 360 < r_angles[1]):
                        # Тригер радар
                        self.agent.radar[i] = 1.0
```

Цей код створює копію точки виходу з лабіринту і повертає її щодо курсу та положення агента в глобальній системі координат. Потім цільова точка транслюється в локальну систему координат агента, що досліджує лабіринт; агент перебуває на початку координат. Після цього ми обчислюємо кут вектору, утвореного від початку координат до цільової точки в локальній системі координат агента. Цей кут є азимутом до точки виходу з лабіринту поточної позиції агента. Коли азимутальний кут знайдено, ми перераховуємо зареєстровані радарні датчики, щоб знайти той, у якого поточний азимутальний кут потрапляє в поле зору (FOV). Відповідний радарний датчик активується шляхом встановлення його значення в 1.0, тоді як інші радарні датчики деактивуються через обнулення їхніх значень.

Оновлення позиції агента. Положення агента в лабіринті необхідно оновлювати на кожному етапі моделювання після отримання відповідних сигналів управління від ШНМ контролера. Для оновлення позиції агента виконується наступний код:

```
def update(self, control_signals):
        if self.exit_found:
                return True # Вихід знайдено
        self.apply_control_signals(control_signals)
        vx = math.cos(geometry.deg_to_rad(self.agent.heading)) * self.agent.speed
        vy = math.sin(geometry.deg_to_rad(self.agent.heading)) * self.agent.speed
        self.agent.heading += self.agent.angular_vel
        if self.agent.heading > 360:
                self.agent.heading -= 360
        elif self.agent.heading < 0:
                self.agent.heading += 360
        new_loc = geometry.Point(
                x = self.agent.location.x + vx,
                y = self.agent.location.y + vy)
        if not self.test_wall_collision(new_loc):
                self.agent.location = new_loc
        self.update_rangefinder_sensors()
        self.update_radars()
        distance = self.agent_distance_to_exit()
        self.exit_found = (distance < self.exit_range)
        return self.exit_found
```

Функція *update(self, control_signals)* визначена у класі *MazeEnvironment* викликається на кожному кроці моделювання. Вона отримує список з керуючими сигналами як вхідні дані і повертає логічне значення, що вказує, чи досяг агент зони виходу після оновлення своєї позиції.

Код на початку цієї функції застосовує отримані сигнали, що управляють, до поточних значень кутової та лінійної швидкостей агента наступним чином:

```
self.agent.angular_vel += (
                        control_signals[0] - 0.5)
self.agent.speed += (control_signals[1] - 0.5)
```

Потім обчислюються компоненти швидкості *x* і *y* разом із напрямом умовної передньої частини агента та використовуються для оцінки його нового положення в лабіринті. Якщо ця нова позиція не стикається ні з однією зі стін лабіринту, то вона призначається агенту і стає його поточною позицією:

```
vx = math.cos(geometry.deg_to_rad(
```

```
        self.agent.heading)) * self.agent.speed
vy = math.sin(geometry.deg_to_rad(
        self.agent.heading)) * self.agent.speed
self.agent.heading += self.agent.angular_vel
if self.agent.heading > 360:
        self.agent.heading -= 360
elif self.agent.heading < 0:
        self.agent.heading += 360
new_loc = geometry.Point(
        x = self.agent.location.x + vx,
        y = self.agent.location.y + vy)
if not self.test_wall_collision(new_loc):
        self.agent.location = new_loc
```

Далі нова позиція агента використовується у функціях, які оновлюють далекомірні та радарні датчики, отримуючи значення нових входів датчиків для наступного тимчасового кроку:

```
self.update_rangefinder_sensors()
self.update_radars()
```

Нарешті наступний блок коду перевіряє, чи досяг агент виходу з лабіринту, що визначається круглою зоною навколо точки виходу з радіусом, рівним значенню поля *exit_range*:

```
distance = self.agent_distance_to_exit()
self.exit_found = (distance < self.exit_range)
return self.exit_found
```

Якщо вихід з лабіринту було досягнуто, значення поля *exit_found* встановлюється рівним *True*, щоб повідомити про успішне вирішення завдання, і це значення повертається з виклику функції.

Зберігання записів агента. Після завершення експерименту нам знадобиться оцінка та візуалізація того, як кожен окремий агент-вирішувач працював протягом еволюційного процесу в усіх поколіннях. Для цього ми збираємо додаткові статистичні дані про кожного агента після запуску моделі проходження лабіринту протягом певної кількості тимчасових кроків. Це досягається шляхом збору додаткових статистичних даних про кожного агента після запуску задачі моделювання вирішення лабіринту протягом визначеної кількості часових кроків.

Колекція записів агента опосередковується двома класами: *AgentRecord* та *AgentRecordStore*.

Клас *AgentRecord* складається з декількох полів даних, як видно з конструктора класу:

```
def __init__(self, generation, agent_id):
        self.generation = generation
        self.agent_id = agent_id
        self.x = -1
        self.y = -1
        self.fitness = -1
        self.hit_exit = False
        self.species_id = -1
        self.species_age = -1
```

Поля мають таке призначення:

- generation містить ідентифікатор покоління, коли було створено запис агента;
- *agent_id* – унікальний ідентифікатор агента;
- *x* та *y* – позиція агента в лабіринті після завершення симуляції;
- *fitness* – підсумкова оцінка пристосованості агента;
- *hit_exit* – це прапор, який вказує, чи досяг агент ділянки виходу з лабіринту;
- *species_id* й *species_age* - ідентифікатор і вік виду, до якого належить агент.

Клас *AgentRecordStore* містить список *AgentRecord* та надає функції для збереження зібраних записів у певний файл та читання з нього. Записи, зібрані під час експерименту, будуть збережені у файлі *data.pickle* в каталозі *output* і можуть бути далі використані для візуалізації працездатності всіх оцінених агентів.

Візуалізація записів агента. Після того як в ході нейроеволюційного процесу будуть зібрані всі оціночні записи всіх агентів, нам буде корисно візуалізувати записані дані, щоб отримати уявлення про стан справ у нашій еволюції. Візуалізація повинна включати кінцеві позиції всіх ви-

рішальних агентів і дозволяти встановлювати порогове значення пристосованості виду для контролю за тим, які види будуть додані до відповідної ділянки. Ми вирішили подати зібрані записи агентів на двох графіках, поданих один над одним. Верхній графік призначений для записів агентів, які належать до видів, у яких показник пристосованості більший або дорівнює вказаному граничному значенню, а нижній графік – для інших записів (дивись рис. 7).

4. Визначення цільової функції з використанням показника пристосованості.

Наразі розглянемо створення успішних агентів, які проходять лабіринт, використовуючи цілеорієнтовану цільову функцію (*goal-oriented objective function*) для управління еволюційним процесом. Цільова функція цього типу заснована на оцінці показника пристосованості вирішувача лабіринту шляхом вимірювання відстані між кінцевим положенням і метою (виходом з лабіринту) після виконання 400 кроків моделювання. Іншими словами, цілеспрямована цільова функція орієнтована на конкретну мету і залежить від кінцевої мети експерименту – досягнення ділянки виходу з лабіринту.

Цілеспрямована цільова функція, задіяна в цьому експерименті, визначається наступним чином. По-перше, нам потрібно визначити функцію помилки як евклідову відстань між кінцевою позицією агента в кінці симуляції та позицією виходу з лабіринту:

$$L = \sqrt{\sum_{i=1}^{2}(a_i - b_i)^2}\,, (1)$$

Де L – функція помилки, a – координати кінцевої позиції агента та b – координати виходу з лабіринту. У цьому експерименті ми використовуємо двовимірний лабіринт, тому координати мають два значення, по одному для кожного вимірювання.

Маючи функцію помилки, ми можемо визначити функцію пристосованості:

$$F = \begin{cases} 1.0 & L \le R_{exit} \\ F_n & otherwise \end{cases}, (2)$$

Де R_{exit} – радіус зони виходу навколо точки виходу з лабіринту, а Fn – нормалізований показник пристосованості, який визначається так:

$$F_n = \frac{D_{init} - L}{D_{init}}, (3)$$

Де D_{init} - це початкова відстань від вирішального агента до виходу з лабіринту на початку навігаційного моделювання.

Рівняння (3) нормалізує показник пристосованості, щоб він розташовувався в діапазоні (0, 1], але може повернути негативні значення в тих поодиноких випадках, коли кінцева позиція агента знаходиться далеко і від його початкової позиції, і від виходу з лабіринту. Щоб уникнути негативних значень, ми будемо застосовувати до нормалізованого показника пристосованості такі поправки:

$$F_n = \begin{cases} 0.01 & F_n \le 0 \\ F_n & otherwise \end{cases}, (4)$$

Коли показник пристосованості менше або дорівнює 0, йому буде присвоєно мінімальне підтримуване значення 0,01; в іншому випадку він залишиться як є. Ми обрали мінімальний показник пристосованості вищий за нуль, щоб надати кожному геному шанс на розмноження.

Наступний код *Python* реалізує описану вище цілеспрямовану цільову функцію:

```
fitness = env.agent_distance_to_exit()
if fitness <= self.exit_range:
        fitness = 1.0
else:
        fitness = (env.initial_distance - fitness) / env.initial_distance
        if fitness <= 0.01:
                fitness = 0.01
```

Далі ми розглянемо проведений експеримент та проаналізуємо отримані результати.

5. Експеримент із простою конфігурацією лабіринту

Отже, ми починаємо наш експеримент зі створення успішного навігаційного агента, який досліджує просту конфігурацію лабіринту. Подібна конфігурація лабіринту, не зважаючи на згадані раніше глухі кути з оманливими локальними оптимумами, має відносно прямий шлях від початкової точки до точки виходу.

На рис. 3 зображена конфігурація простого лабіринту, задіяного в поточному експерименті.

Лабіринт має дві фіксовані позиції, подзначені забарвленими кругами. Верхнє ліве коло позначає початкову позицію агента-вирішувача лабіринту. Нижнє праве коло позначає точне місце виходу з лабіринту, яке має бути знайдене вирішувачем. Для виконання завдання, розв'язувач лабіринту повинен досягти поблизу точки виходу з лабіринту, позначеної спеціальною зоною виходу навколо нього.

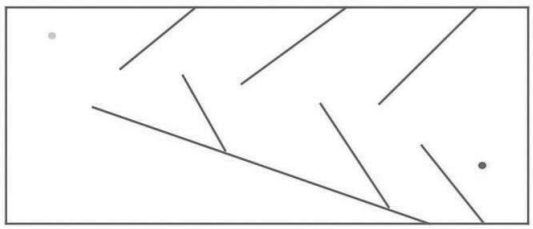

Рис. 3. Конфігурація простого лабіринту

Вибір гіперпараметрів. Згідно з визначенням цільової функції (формула 3), максимальне значення показника пристосованості агента, яке може бути отримано при досягненні заданого околу виходу з лабіринту, становить 1.0. Завдання пошуку рішення для задачі навігації лабіринту є доволі складним, тому для успішного пошуку рішення необхідно використовувати широку зону пошуку у просторі рішень. Для цього, методом спроб і помилок ми виявили, що чисельність популяції може бути рівною 250. Це дає гарний результат, одночасно маючи адекватний час на завершення алгоритму нейроеволюції.

Початкова конфігурація фенотипу ШНМ має 10 вхідних вузлів, 2 вихідних вузли та 1 прихований вузол. Вузли входу та виходу відповідають вхідним датчикам та виходам керуючого сигналу. Прихований вузол спрямований на введення нелінійності від початку нейроеволюційного процесу та економію часу еволюції на додавання цього вузла.

Для розширення зони пошуку рішень нам також необхідно збільшити видове розмаїття популяції, щоб спробувати різні зміни геному протягом обмеженого числа поколінь. Цього можна досягти зменшенням порогу сумісності, або через збільшення значень коефіцієнтів, які використовуються для розрахунку показників сумісності геному. У цьому експерименті ми використовували обидві поправки, тому що ландшафт функції пристосованості оманливий, і нам потрібно підкреслити навіть крихітні зміни в конфігурації геному, щоб створити новий вид. На це впливають такі параметри конфігурації:

```
[NEAT]
compatibility_disjoint_coefficient = 1.1
[DefaultSpeciesSet]
compatibility_threshold = 3.0
```

Ми особливо зацікавлені у створенні оптимальної конфігурації ШНМ, яка має мінімальну кількість прихованих вузлів та зв'язків. Оптимальна конфігурація менш затратна в обчислювальному відношенні під час навчання нейроеволюційним методом, а також під час фази виведення у симуляторі проходження лабіринту. Оптимальну конфігурацію нейромережі можна отримати, зменшивши можливість додавання нових вузлів, як показано в наступному фрагменті з файлу конфігурації NEAT:

```
node_add_prob = 0.1
node_delete_prob = 0.1
```

Нарешті ми дозволяємо нейроеволюційному процесу використовувати не тільки нейромережу з прямим зв'язком, а й рекурентні нейромережі. Дозволяючи рекурентні з'єднання, ми даємо можливість нейромережі мати пам'ять і стати кінцевим автоматом. Це корисно для еволюційного процесу. Наступний гіперпараметр конфігурації впливає на цей фактор:

```
feed_forward = False
```

Гіперпараметри, наведені вище, виявилися корисними для алгоритму NEAT, який використовується в експерименті для навчання протягом обмеженої кількості поколінь успішного агента, що проходить лабіринт.

Результати експерименту. Експеримент був проведений з використанням Python 3.10 та бібліотеки NEAT-Python версії 0.92 (10). Робоча станція, що використовувалась для цього, має наступні параметри: CPU 2,3 GHz 8-Core Intel Core i9, 16 GB 2667 MHz DDR4, macOS 13.5.2.

Успішний агент-вирішувач був знайдений після 144 поколінь нейроеволюції та має наступну конфігурацію геному:

Nodes:

0 DefaultNodeGene(key=0, bias=5.534849614521037, response=1.0, activation=sigmoid, aggregation=sum)

1 DefaultNodeGene(key=1, bias=1.8031133229851957, response=1.0, activation=sigmoid, aggregation=sum)

158 DefaultNodeGene(key=158, bias=-1.3550878188609456, response=1.0, activation=sigmoid, aggregation=sum)

Connections:

DefaultConnectionGene(key=(-10, 158), weight=-1.6144052085440168, enabled=True)

DefaultConnectionGene(key=(-8, 158), weight=-1.1842193888036392, enabled=True)

DefaultConnectionGene(key=(-7, 0), weight=-0.3263706518456319, enabled=True)

DefaultConnectionGene(key=(-7, 1), weight=1.3186165993348418, enabled=True)

DefaultConnectionGene(key=(-6, 0), weight=2.0778575294986945, enabled=True)

DefaultConnectionGene(key=(-6, 1), weight=-2.9478037554862824, enabled=True)

DefaultConnectionGene(key=(-6, 158), weight=0.6930281879212032, enabled=True)

DefaultConnectionGene(key=(-4, 1), weight=-1.9583885391583729, enabled=True)

DefaultConnectionGene(key=(-3, 1), weight=5.5239054588484775, enabled=True)

DefaultConnectionGene(key=(-1, 0), weight=0.04865917999517305, enabled=True)

DefaultConnectionGene(key=(158, 0), weight=0.6973191076874032, enabled=True)

Успішний агент-вирішувач зміг досягти зони виходу з лабіринту за 388 кроків з відведених 400. Конфігурація нейромережі успішного вирішувача містить 2 вихідні вузли та 1 прихований вузол, з 11 зв'язками між цими вузлами та входами. Остаточна конфігурація нейромережі показана на рис. 4.

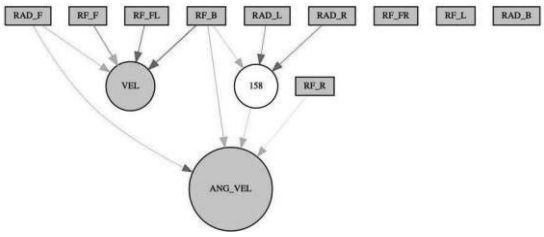

Рис. 4. Конфігурація нейромережі, що відповідає успішному вирішувачу простого лабіринту

Цікаво поглянути на граф нейромережі з точки зору впливу різних входів датчиків на вихідні керуючі сигнали. Ми можемо бачити, що конфігурація нейромережі повністю ігнорує вхідні сигнали від переднього і лівого далекомірних датчиків (*RF_FR* і *RF_L*) і від секторного радарного датчика *RAD_B* робота. Водночас лінійні та кутові швидкості робота залежать від унікальних комбінацій інших датчиків.

Крім того, ми можемо бачити агрегацію лівого та правого радарних датчиків (*RAD_L* і *RAD_R*) з далекоміром *RF_B* через прихований вузол, який потім ретранслює агрегований сигнал вузлу, що управляє кутовою швидкістю. Якщо ми подивимося зображення простої конфігурації лабіринту (рис. 3), то подібна агрегація виглядає досить природною. Вона дозволяє

роботу розвернутися і продовжити досліджувати лабіринт, коли він застряг в одному з глухих кутів, де розташовані локальні оптимуми.

Оцінку пристосованості агентів за поколіннями показано на рис. 5. На цьому графіку ми можемо бачити, що еволюційний процес зміг продукувати досить успішних агентів-вирішувачів у поколінні 44 з оцінкою пристосованості 0,96738. Але знадобилося ще 100 поколінь, аби розвинути геном, який кодує нейромережу успішного агента.

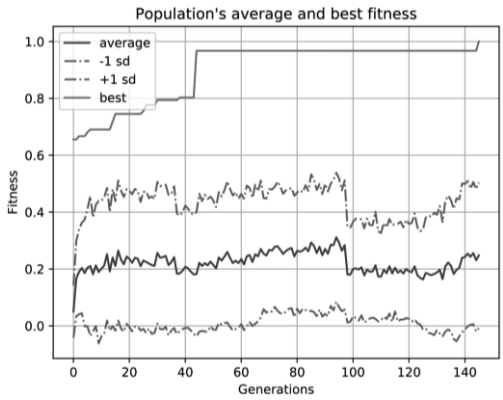

Рис. 5. Середні оцінки пристосованості агентів за поколіннями

Крім того, цікаво відзначити, що підвищення продуктивності в поколінні 44 генерується видами з ID 1, але геном успішного вирішувача належить виду з ID 7, який навіть не був відомий під час першого сплеску. Види, що породили чемпіона, з'явилися через 12 поколінь і залишалися в популяції до кінця, зберігаючи корисну мутацію і вдосконалюючись на її основі.

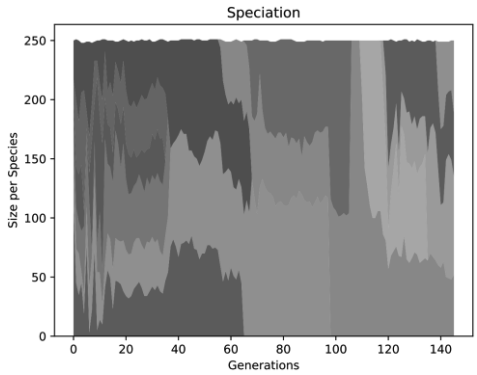

Рис. 6. Видоутворення за поколіннями

Видоутворення протягом кількох поколінь показано на рис. 6. На цьому графіку ми також спостерігаємо вид з ID 7. Цей вид у кінцевому підсумку породив геном успішного вирішувача за час еволюційного процесу. Розмір виду 7 значно варіюється протягом всього його життя, і свого часу він був єдиним видом у всій популяції упродовж декількох поколінь (від 105 до 108).

Візуалізація запису агента. У цьому експерименті представлено новий метод візуалізації, який дозволяє візуально розрізняти ефективність різних видів в еволюційному процесі. Візуалізація базується на збережених даних щодо проходження лабіринту кожним з агентів-вирішувачів протягом усіх епох нейроеволюції.

Візуалізатор малює кінцеві позиції агентів на карті лабіринту в кінці симуляції проходження. Кінцева позиція кожного агента представлена у вигляді кольорового кола. Колір кола означає вид, до якого належить конкретний агент. Кожен вид, отриманий у процесі еволюції, має унікальний колірний код. Результати цієї візуалізації показано на рис. 7.

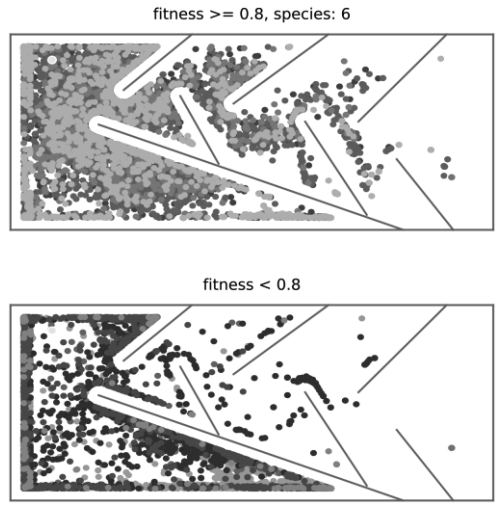

Рис. 7. Візуалізація оцінки агентів-вирішувачів

Для більшої інформативності візуалізації введено поріг пристосування для фільтрації найбільш ефективних видів. Верхня половина рисунка показує кінцеві

позиції вирішальних агентів, що належать до видів-чемпіонів (показник пристосованості вище 0,8). Як можна побачити, організми, що належать до цих шести видів, є активними дослідниками, у яких є гени, що провокують пошук у невідомих місцях лабіринту. Їхні кінцеві розташування майже рівномірно розподілені ділянками лабіринту навколо початкової точки, мають низьку щільність у глухих кутах локальних оптимумів.

Водночас на нижній половині рисунка можна побачити, що еволюційні невдахи демонструють консервативнішу поведінку, концентруючись в основному біля стін у стартовій зоні і в найбільш вираженій ділянці локального оптимуму - найбільшому глухому куті, який знаходиться в нижній частині лабіринту.

Висновки

Досліджено можливість навчання ефективних агентів-розв'язувачів задачі навігації лабіринтом за допомогою алгоритму NEAT. Надано математичний аналіз цільової функції, яка підходить для оптимізації процесу навчання агентів-розв'язувачів під час нейроеволюції. За допомогою запропонованої цільової функції було створено програмний продукт для управління нейроеволюційним процесом.

Було розроблено систему моделювання поведінки робота, який може автономно знайти вихід з лабіринту, використовуючи сигнали від різних типів вхідних датчиків. За допомогою розробленої моделюючої системи, було здійснено низку експериментів для встановлення оптимальних значень гіперпараметрів для ефективного навчання керуючої ШНМ методом нейроеволюції.

Було створено спеціалізоване програмне рішення для відображення процесу навчання агентів-розв'язувачів. Розроблені методи відображення суттєво сприяють пошуку оптимальних параметрів алгоритму NEAT за рахунок наочної демонстрації впливу зміни того чи іншого параметру на процес навчання.

У майбутньому отримані дані можуть сприяти створенню енергозберігаючих керуючих ШНМ для регулювання роботи фізичних роботів. Крім того, розроблена система комп'ютерної імітації, дозволяє виконувати велику кількість досліджень у стислі терміни та з мінімальними витратами.

Все це обумовлює актуальність досліджень щодо використання алгоритму NEAT для вирішення широкого кола прикладних задач та проблем моделювання автономних агентів.

References

1. Jean-Baptiste Mouret and Stephane Doncieux. 2012. Encouraging behavioral diversity in evolutionary robotics: An empirical study. Evolutionary computation 20, 1 (2012), 91–133.
2. Joel Lehman and Kenneth O Stanley. 2010. Revising the evolutionary computation abstraction: minimal criteria novelty search. In Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2010). ACM, 103–110.
3. Joel Lehman and Kenneth O Stanley. 2011. Abandoning objectives: Evolution through the search for novelty alone. Evolutionary computation 19, 2 (2011), 189–223.
4. Kenneth O Stanley and Risto Miikkulainen. 2002. Evolving neural networks through augmenting topologies. Evolutionary computation 10, 2 (2002), 99–127.
5. Justin K Pugh, Lisa B Soros, and Kenneth O Stanley. 2016. Quality diversity: A new frontier for evolutionary computation. Frontiers in Robotics and AI 3 (2016), 40.
6. Jonathan C. Brant and Kenneth O. Stanley. 2017. Minimal criterion coevolution: a new approach to open-ended search. In Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '17). Association for Computing Machinery, New York, NY, USA, 67–74.
7. Iaroslav Omelianenko. Hands-On Neuroevolution with Python: Build high-performing artificial neural network architectures using neuroevolution-based algorithms. Birmingham, UK: Packt Publishing Ltd, 2019. ISBN: 9781838824914, 368 pp.
8. Iaroslav Omelianenko. Creation of Autonomous Artificial Intelligent Agents Using Novelty Search Method of Fitness Function Optimization. NewGround LLC, Sept. 2018, https://hal.science/hal-01868756.
9. Iaroslav Omelianenko. “Autonomous Artificial Intelligent Agents”. In: Machine Learning

and the City. John Wiley Sons, Ltd, 2022. Chap. 12, pp. 263–285. ISBN: 9781119815075, DOI: 10.1002/9781119815075.ch21, URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/9781119815075.ch21, SCOPUS: 2-s2.0-85147956837, https://www.scopus.com/record/display.uri?eid=2-s2.0-85147956837&origin=inward&txGid=b119d677e846feb8f319ba241f759c75

10. McIntyre, A., Kallada, M., Miguel, C. G., Feher de Silva, C., & Netto, M. L. neat-python [Computer software], https://github.com/CodeReclaimers/neat-python

Одержано: 28.09.2023

Про автора:

Омельяненко Ярослав Вікторович,
молодший науковий співробітник.
Кількість зарубіжних публікацій – 7.
Кількість патентів - 2.
https://orcid.org/0000-0002-2190-5664

Місце роботи автора:

Інститут Програмних Систем
НАН України,
03187, м. Київ-187,
проспект Академіка Глушкова, 40.
Тел.: (044) 526 3559.
E-mail: yaric@newground.com.ua

УДК 51:681.3.0 http://doi.org/10.15407/pp2023.04.090

А.Ю. Дорошенко, Р.В. Кушніренко

РЕКУРЕНТНІ НЕЙРОННІ МЕРЕЖІ ДЛЯ ЗАДАЧІ УТОЧНЕННЯ ЧИСЕЛЬНИХ МЕТЕОРОЛОГІЧНИХ ПРОГНОЗІВ

Зроблено короткий огляд застосування "глибокого навчання" до геонаукових задач. Порівняні два найпопулярніші види архітектур рекурентних нейронних мереж, а саме мережу довгої короткочасної пам'яті та вентильний рекурентний вузол. Показано, що усі моделі з вентильними рекурентними вузлами є більш ефективними за моделі довгої короткочасної пам'яті. На основі доступних даних спостережень здійснено чисельні експерименти з уточнення прогнозу за допомогою машинного навчання. Виявлено, що кращою архітектуру рекурентних нейронних мереж для розв'язання задачі уточнення чисельних метеорологічних прогнозів є вентильний рекурентний вузол.
Ключові слова: "глибоке навчання", рекурентні нейронні мережі, метеорологічне прогнозування.

Вступ

Протягом останнього десятиліття "глибоке навчання" стало важливою частиною дослідницьких і оперативних схем геонаукової обробки, що стосуються атмосфери, поверхні суші та океану. Цьому сприяли, поміж іншого, збільшення доступності даних спостережень, а також підвищення швидкості їхньої передачі, що вже перевищує сотні терабайт на день [1]. Ці дані надходять від безлічі датчиків, що вимірюють різні часово і просторово інтегровані величини. Зокрема, вони включають дані дистанційного зондування на висоті від кількох метрів до сотень кілометрів над Землею, а також спостереження на місці (на поверхні та під нею) за допомогою автономних датчиків.

Хоча "глибоке навчання" досягло помітних успіхів у моделюванні впорядкованих послідовностей і даних із просторовим контекстом у сферах комп'ютерного зору, систем розпізнавання мови та керування [2], а також у таких наукових галузях як фізика [3], хімія [4] та біологія [5], його застосування до проблем геонауки знаходиться в зародковому стані. Однак деякі спроби його застосування до таких ключових проблем як класифікація, регресія, виявлення аномалій та прогнозування залежного від простору або часу стану мають перспективні рішення. До прикладу, є декілька досліджень, що демонструють застосування "глибокого навчання" до проблеми прогнозування екстремальних погодних умов [6,7]. Зауважимо, що ця задача є проблематичною для традиційного машинного навчання. Згадані дослідження свідчать про успіх у застосуванні архітектур "глибокого навчання" до виокремлення просторових і часових характеристик для визначення та класифікації екстремальних ситуацій (наприклад, штормів) у вихідних даних числової моделі прогнозування погоди. Такий підхід дозволяє швидко виявляти такі події та моделювати прогнози без використання суб'єктивних анотацій людини або методів, які покладаються на заздалегідь визначені порогові значення для швидкості вітру та інших метеорологічних величин.

Нагадаємо, що підходи "глибокого навчання" класично поділяються на просторові (наприклад, згорткові нейронні мережі [8] для класифікації об'єктів) і послідовні (наприклад, рекурентні нейронні мережі для розпізнавання мовлення [9]). Згорткові мережі являють собою стек фільтрів малого розміру з невеликою кількістю параметрів, які власне "навчаються". Вони застосовуються до зображень або інших даних на прямокутній сітці для отримання узагальнених характерних особливостей досліджуваного об'єкта. У царині геонауки згорткові мережі можуть бути використані для виявлення просторових характеристик, наприклад, під час аналізу супутникових зображень [10]. На противагу рекурентні нейронні мережі були розроблені для вивчення залежних від часу особливостей

даних. Рекурентність — це лише загальна ідея, яка полягає у тому, що топологічно така архітектура нейронної мережі може бути представлена орієнтованим у часі графом. Завдяки цьому утворюється "пам'ять" мережі (вектор внутрішнього стану), що і дозволяє виявляти динамічні (у часовому вимірі) характеристики досліджуваних даних.

Однак спостерігається все більша зацікавленість у поєднанні цих двох підходів. Прототипним прикладом цього поєднання є прогнозування відео та руху [11], проблема, яка має разючу подібність до багатьох динамічних геонаукових проблем. Тут ми стикаємося з багатовимірними структурами, змінними в часі. Наприклад, параметри рослинного покриву, що впливають на вуглецевий цикл та випаровування. Вже існують дослідження, що починають застосовувати комбіновані згортково-рекурентні підходи до таких геонаукових проблем як прогнозування опадів [12]. Моделювання динаміки атмосфери та океану, моделювання поширення вогню чи руху ґрунту також є прикладами проблем, де важлива просторово-часова динаміка. Але наразі вони не отримали переваг від застосування комбінованих згортково-рекурентних підходів "глибокого навчання".

Коротко кажучи, подібність між типами даних, притаманних класичним застосуванням "глибокого навчання", і даних, з якими працює геонаука, є переконливим аргументом на користь інтеграції "глибокого навчання" в геонауки. Зображення є аналогом двовимірних полів даних, що містять певні змінні за аналогією з триплетами кольорів (значення RGB) на фотографіях, тоді як відео можна пов'язати з послідовністю зображень, тобто з двовимірними полями, які змінюються у часі. Подібним чином природна мова та мовлення мають такі ж характерні особливості динамічних часових рядів, що їх мають дані, притаманні геонауковій сфері. Крім того, класифікація, регресія, виявлення аномалій і динамічне моделювання є типовими проблемами як для класичних застосувань "глибокого навчання", так і для геонаук.

Проте, як було показано у [13], ще рано говорити про повний перехід геонауки, зокрема, сфери метеорологічного прогнозування, на методи, що базуються лише на "глибокому навчанні" і спостережуваних даних. Як було зазначено вище, попри те, що "глибоке навчання" останнім часом показує себе успішно у різних сферах, і попри те, що робляться спроби його застосування до метеорологічних задач, такі дослідження все ще знаходяться в зародковому стані. Це пов'язано передусім з тим, що характерні особливості метеорологічних даних вимагають розробки нових підходів поза межами класичних концепцій комп'ютерного зору, розпізнавання мовлення та інших типових задач, поставлених перед "глибоким навчанням". На відміну від них, підвищення точності прогнозування є хоча і дуже важливим, однак недостатнім компонентом. Дуже важливим складником тут є також надання можливості інтерпретації та розуміння результатів, включаючи їх візуалізацію для аналізу людьми. А, як відомо, інтерпретованість була визначена як потенційна слабкість "глибоких" нейронних мереж, і досягнення цієї мети зараз є центральною проблемою для "глибокого навчання" [14]. Ця галузь все ще є далекою від створення зрозумілих моделей, а також від надання можливості визначення причин закономірностей на основі даних спостережень [15]. Однак ми маємо визнати, що на практиці, враховуючи складність сучасних моделей чисельного метеорологічного прогнозування, також нелегко відстежити зв'язок між результатами їхньої роботи та припущеннями, на яких їх побудовано, а це, очевидно, обмежує їхню інтерпретованість. Крім цього, моделі "глибокого навчання" можуть бути фізично непослідовними або неправдоподібними, хоча їхня статистична точність буде високою. Це може відбуватися через надмірну екстраполяцію та/або статистичну упередженість спостережень. Інтеграція знань предметної області та досягнення фізичної узгодженості моделей за допомогою навчання відповідно до законів фізики, може забезпечити дуже сильні теоретичні обмеження на додаток до спостережуваних даних. Треба зазначити, що робота над підходами, що розв'язували б ці та інші проблеми, триває, і навіть деякі проблеми

тією чи іншою мірою вже розв'язані, та попри це не існує єдиного методу, що розв'язував би усі проблеми одночасно, а саме він і потрібен аби успішно здійснити повний перехід метеорологічного прогнозування на методи "глибокого навчання".

Однак відповідно до [16], є тенденція до розробки гібридних підходів моделювання, які поєднали б моделі фізичних процесів з універсальністю інструментів "глибокого навчання" для досягнення кращих результатів. Зокрема, існують дослідження можливості успішного використання "глибокого навчання" на різних стадіях чисельного прогнозу: обробка спостережень [17], асиміляція даних [18], прогностична модель [19] та постпроцесинг отриманих результатів [20].

Власне, дана стаття присвячена застосуванню "глибокого навчання" до постпроцесингу результатів прогнозу приземної температури, отриманого за допомогою чисельних гідродинамічних методів метеорологічного прогнозування, а саме порівнянню двох найпопулярніших видів рекурентних нейронних мереж у застосуванні до даної прикладної задачі. Це порівняння дасть змогу виявити оптимальну архітектуру нейронної мережі для подальшого досягнення належного ступеня точності метеорологічних прогнозів.

Опис даних

"Глибоке навчання" як техніка виокремлення характерних особливостей даних суттєво залежить від якості, репрезентативності та цілісності використовуваних даних. Тому правильний відбір і підготовка даних є важливими факторами для отримання хороших і узагальнюючих результатів.

Зокрема, відбір даних має бути спрямований на охоплення якнайбільше повної варіативності значень змінних, на яких базуватиметься власне навчання нейромережевої моделі. Хороші дані мають дозволяти моделі охоплювати зв'язки між змінними, на основі яких робиться прогноз. Водночас важливим є уникнення надлишковості у даних.

Нижче поданий опис даних, використаних для дослідження, описаного у даній статті. Ці дані складалися з чотириелементних кортежів і містили наступну інформацію:

- дата,
- час за Гринвічем,
- прогнозоване значення температури (Fcst), завчасністю в одну добу від моменту ініціалізації чисельної регіональної моделі,
- спостережуване значення температури (Obs).

Чисельною моделлю прогнозу погоди, результат роботи якої ми хочемо покращити, є модель однойменного європейського консорціуму COSMO (Consortium for Small-scale Modelling). Ця модель використовується в Українському гідрометеорологічному інституті ДСНС України та НАН України для наукових та прикладних задач, починаючи із липня 2011 р. [21]. Нагадаємо, що COSMO є негідростатичною моделлю, яка здатна ефективно відтворювати широкий спектр атмосферних процесів у масштабі мезо-β та мезо-γ. В основу динамічного ядра моделі покладено рівняння термо- та гідродинаміки, що описують потік у вологій атмосфері. Різноманітні фізичні процеси враховуються схемами параметризації [22].

Рис. 1 зображує розрахункову область чисельної регіональної моделі: кількість вузлів із заходу на схід – 209; кількість вузлів із півдня на північ – 101; кількість рівнів по вертикалі – 50; крок ~ 14 км.

Рис.1. Розрахункова область моделі прогнозу погоди COSMO

Наявні дані охоплюють проміжок часу від 01.07.2012 до 31.03.2014, або 639 днів. Спостереження проводилися кожні три години, а саме о 00:00, 03:00, 06:00, 09:00, 12:00, 15:00, 18:00 і 21:00 за Гринвічем. Для цих же моментів часу обчислювався і прогноз регіональної моделі.

Таким чином, для кожної дати маємо по вісім кортежів. Відповідно 639 днів дають 5112 кортежів.

Що до просторової приналежності, то дані охоплюють спостережувані значення і прогнози для станцій "Біла Церква", "Бориспіль", "Київ", "Миронівка", "Тетерів", "Фастів", "Чорнобиль" та "Яготин".

Рекурентні нейронні мережі

Як зазначалося вище, поточне дослідження має за мету порівняння двох найпопулярніших видів архітектур рекурентних нейронних мереж у застосуванні до задачі постпроцесингу результатів прогнозу приземної температури, отриманого за допомогою чисельних гідродинамічних методів метеорологічного прогнозування.

Зауважимо, що використання концепції рекурентних нейронних мереж зумовлене тим, що метеорологічні дані мають природу часового ряду, тобто містять залежні від часу особливості. А рекурентні нейронні мережі були розроблені саме для роботи з даними такого роду.

Двома найпопулярнішими видами рекурентних нейронних мереж, про які йшлося вище, є мережа довгої короткочасної пам'яті (long short-term memory, LSTM) [23] та вентильний рекурентний вузол (gated recurrent unit, GRU) [24]. Ці методи завдячують своєю популярністю тому, що дозволяють уникнути основних проблем, які виникають під час аналізу довгострокових залежностей. Цими проблемами є ефект зникнення градієнтів (vanishing gradient effect) і безпосередньо пов'язаний з ним ефект вибуху градієнтів (exploding gradients effect).

Наведемо теоретичні відомості, що стосуються згаданих вище архітектур нейронних мереж.

Вузол GRU працює наступним чином. У кожний момент часу йому на вхід подаються вхідний вектор x_t і вихідний вектор з попереднього моменту часу h_{t-1}. Вихідний вектор h_t обчислюється як лінійна інтерполяція між h_{t-1} і поточним кандидатом $\hbar_t$:

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \hbar_t,$$

де z_t – вектор вузла уточнення. Він визначає, які долі першого і другого векторів впливатимуть на поточне значення. $\odot$ позначає добуток Адамара.

Вектор вузла уточнення обчислюється наступним чином:

$$z_t = \sigma_g(W_z x_t + U_z h_{t-1} + b_z).$$

Поточний кандидат $\hbar_t$ обчислюється подібно до стандартної рекурентної нейронної мережі:

$$\hbar_t = \sigma_h(W_h x_t + U_h(r_t \odot h_{t-1}) + b_h),$$

де r_t – вектор вузла скидання, що обчислюється подібно до вектора вузла уточнення:

$$r_t = \sigma_g(W_r x_t + U_r h_{t-1} + b_r).$$

W_z, U_z, W_h, U_h, W_r, W_r позначають матриці параметрів, аb_z, b_h, b_r позначають вектори параметрів. Вектори вузлів уточнення і скидання активуються сигмоїдною функцією, а вектор поточного кандидата – гіперболічним тангенсом.

Що до LSTM, то ця архітектура являє собою дещо ускладнену версію GRU. Замість двох вузлів (уточнення та скидання) вона має три — вузли входу, забування та виходу. Звичайно, це робить мережі довгої короткочасної пам'яті повільнішими для навчання та використання, але разом з тим, вони можуть бути ефективнішими, коли йдеться про зберігання та доступ до довгострокових залежностей.

Зауважимо, що і мережі довгої короткочасної пам'яті, і вентильні рекурентні вузли можуть розв'язувати широкий спектр задач, зокрема, розпізнавання мовлення, машинний переклад та прогнозування часових рядів. Як правило, LSTM-

мережі більш ефективні в задачах, які вимагають зберігання та доступ до довгострокових залежностей. З іншого боку, GRU-мережі ефективніші в задачах, які вимагають швидкого навчання та адаптації до нових вхідних даних. Проте варто пам'ятати, що не існує єдиного найбільш ефективного типу рекурентних нейронних мереж, який підходив би для усіх завдань. Тому вибір між LSTM і GRU залежатиме від конкретних вимог розв'язуваної задачі.

Як правило, доцільно спробувати обидві архітектури та порівняти їхню ефективність у застосуванні до тієї чи іншої конкретної задачі. Власне, це і є мета проведення даного дослідження, а саме порівняння цих двох архітектур у застосуванні до задачі постпроцесингу результатів чисельного метеорологічного прогнозування.

Опис експерименту

Як було зазначено вище, порівняння архітектур рекурентних нейронних мереж буде здійснено у застосуванні до прикладної задачі уточнення прогнозу приземної температури, отриманого за допомогою чисельної моделі прогнозу погоди COSMO [25].

Для кожної метеорологічної станції було натреновано дві нейромережеві моделі (одна з GRU-шаром, інша з LSTM-шаром), які мали б якнайкраще виокремити фізичні особливості конкретного пункту спостереження. Тож, ми маємо 16 натренованих моделей.

Нейромережева топологія та інші конфігураційні параметри були однаковими для усіх моделей. Першим шаром нейронної мережі (після вхідного) було взято рекурентний шар (GRU або LSTM), що складався з 64 вузлів для GRU та 32 вузлів для LSTM, оскільки менша і більша кількість давали гірші результати. Після нього було додано один прихований повнозв'язний шар із 64 вузлами. Останній шар запропонованої архітектури був також повнозв'язним і мав 8 вузлів, оскільки вихідний вектор повинен мати розмірність добового вектора прогнозу (8 значень).

Рис. 2 зображує описану топологію нейромережевих моделей.

Що стосується даних, то для кожної метеорологічної станції уся їх сукупність була розбита на три класи: тренувальні (період з 01.07.2012 до 30.06.2013; 365 днів), валідаційні (період з 01.07.2013 до 31.10.2013; 123 дні) і тестувальні (з 01.11.2013 до 01.04.2014; 151 день).

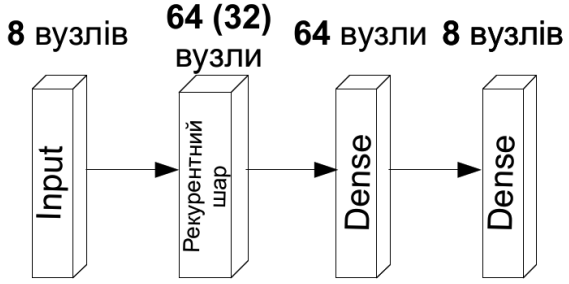

Рис. 2. Запропонована архітектура нейронної мережі

Зауважимо, що поділ на три класи (набори) — це загальна практика для "глибокого навчання" [26]. Тренувальний набір є найбільшим і використовується для оновлення вагових коефіцієнтів моделі шляхом зворотного розповсюдження помилки або інших алгоритмів навчання. Другий набір, валідаційний, використовується виключно для налаштування гіперпараметрів: кількості шарів, типів шарів, функцій активації, цільових функцій, швидкості навчання тощо. Ключовою метою цього налаштування є підвищення здатності мережі до узагальнення для гарантії, що мережа добре функціонуватиме на невідомих для неї даних. Третій набір даних — це тестовий набір, раніше невідомі дані, які використовуються для оцінювання мережі після налаштування.

Зазначимо, що усі експерименти проводилися з використанням відкритого нейромережевого інтерфейсу Keras [27] і відкритої програмної бібліотеки для "глибокого навчання" TensorFlow [28]. Програмний код був написаний мовою Python [29].

Отримані результати

Порівняння ефективностей двох архітектур рекурентних нейронних мереж здійснювалось за допомогою стандартного способу оцінювання похибок, а саме за

допомогою середнього квадратичного відхилення (RMSE). Як бачимо з таблиці, усі моделі з вентильними рекурентними вузлами (GRU) є оптимальнішими за моделі довгої короткочасної пам'яті (LSTM).

Таблиця. Порівняння архітектур рекурентних нейронних мереж

Станція	RMSE (GRU), °C	RMSE (LSTM), °C
Біла Церква	1.8865	2.1451
Бориспіль	1.9958	2.1456
Київ	1.9824	2.1334
Миронівка	2.0099	2.4484
Тетерів	2.0215	2.3027
Фастів	1.8565	2.1394
Чорнобиль	2.0094	2.0355
Яготин	2.1157	2.3793

Висновки

На прикладі прогнозів моделі COSMO приземної температури повітря для восьми метеорологічних станцій Київської області та відповідних їм даних фактичних спостережень було порівняно два найпопулярніші види архітектур рекурентних нейронних мереж, а саме мережу довгої короткочасної пам'яті та вентильний рекурентний вузол.

Показано, що усі моделі з вентильними рекурентними вузлами є ефективнішими за моделі довгої короткочасної пам'яті.

Таким чином, виявлено найкращу архітектуру рекурентних нейронних мереж для розв'язання задачі постпроцесингу результатів чисельного метеорологічного прогнозування.

References

1. Agapiou, A., 2017. Remote sensing heritage in a petabyte-scale: satellite data and heritage Earth Engine© applications. International Journal of Digital Earth, 10(1), pp.85-102.
2. LeCun, Y., Bengio, Y. and Hinton, G., 2015. Deep learning. nature, 521(7553), pp.436-444.
3. Bhimji, W., Farrell, S.A., Kurth, T., Paganini, M., Prabhat and Racah, E., 2018, September. Deep neural networks for physics analysis on low-level whole-detector data at the LHC. In Journal of Physics: Conference Series (Vol. 1085, p. 042034). IOP Publishing.
4. Schütt, K.T., Arbabzadah, F., Chmiela, S., Müller, K.R. and Tkatchenko, A., 2017. Quantum-chemical insights from deep tensor neural networks. Nature communications, 8(1), p.13890.
5. Alipanahi, B., Delong, A., Weirauch, M.T. and Frey, B.J., 2015. Predicting the sequence specificities of DNA-and RNA-binding proteins by deep learning. Nature biotechnology, 33(8), pp.831-838.
6. Liu, Y., Racah, E., Correa, J., Khosrowshahi, A., Lavers, D., Kunkel, K., Wehner, M. and Collins, W., 2016. Application of deep convolutional neural networks for detecting extreme weather in climate datasets. arXiv preprint arXiv:1605.01156.
7. Racah, E., Beckham, C., Maharaj, T., Ebrahimi Kahou, S., Prabhat, M. and Pal, C., 2017. Extremeweather: A large-scale climate dataset for semi-supervised detection, localization, and understanding of extreme weather events. Advances in neural information processing systems, 30.
8. LeCun, Y., Bottou, L., Bengio, Y. and Haffner, P., 1998. Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), pp.2278-2324.
9. Sak, H., Senior, A. and Beaufays, F., 2014. Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition. arXiv preprint arXiv:1402.1128.
10. Zhu, X.X., Tuia, D., Mou, L., Xia, G.S., Zhang, L., Xu, F. and Fraundorfer, F., 2017. Deep learning in remote sensing: A comprehensive review and list of resources. IEEE geoscience and remote sensing magazine, 5(4), pp.8-36.
11. Oh, J., Guo, X., Lee, H., Lewis, R.L. and Singh, S., 2015. Action-conditional video prediction using deep networks in atari

games. Advances in neural information processing systems, 28.

12. Shi, X., Chen, Z., Wang, H., Yeung, D.Y., Wong, W.K. and Woo, W.C., 2015. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. Advances in neural information processing systems, 28.
13. Schultz, M.G., Betancourt, C., Gong, B., Kleinert, F., Langguth, M., Leufen, L.H., Mozaffari, A. and Stadtler, S., 2021. Can deep learning beat numerical weather prediction?. Philosophical Transactions of the Royal Society A, 379(2194), p.20200097.
14. Montavon, G., Samek, W. and Müller, K.R., 2018. Methods for interpreting and understanding deep neural networks. Digital signal processing, 73, pp.1-15.
15. Runge, J., Petoukhov, V., Donges, J.F., Hlinka, J., Jajcay, N., Vejmelka, M., Hartman, D., Marwan, N., Paluš, M. and Kurths, J., 2015. Identifying causal gateways and mediators in complex spatio-temporal systems. Nature communications, 6(1), p.8502.
16. Bauer, P., Dueben, P.D., Hoefler, T., Quintino, T., Schulthess, T.C. and Wedi, N.P., 2021. The digital revolution of Earth-system science. Nature Computational Science, 1(2), pp.104-113.
17. Prudden, R., Adams, S., Kangin, D., Robinson, N., Ravuri, S., Mohamed, S. and Arribas, A., 2020. A review of radar-based nowcasting of precipitation and applicable machine learning techniques. arXiv preprint arXiv:2005.04988.
18. Bonavita, M. and Laloyaux, P., 2020. Machine learning for model error inference and correction. Journal of Advances in Modeling Earth Systems, 12(12), p.e2020MS002232.
19. Krasnopolsky, V.M., Fox-Rabinovitz, M.S. and Chalikov, D.V., 2005. New approach to calculation of atmospheric model physics: Accurate and fast neural network emulation of longwave radiation in a climate model. Monthly Weather Review, 133(5), pp.1370-1383.
20. Rasp, S. and Lerch, S., 2018. Neural networks for postprocessing ensemble weather forecasts. Monthly Weather Review, 146(11), pp.3885-3900.
21. Shpyg, V., Budak, I., Pishniak, D. and Poperechnyi, P., 2013, November. The application of regional NWP models to operational weather forecasting in Ukraine. In CAS Technical Conference (TECO) on" Responding to the Environmental Stressors of the 21st Century" Available from: http://www.wmo.int/pages/prog/arep/cas/documents/Ukraine-NWPModels. pdf [Accessed 27/02/2020].
22. Doms, G. and Baldauf, M., 2011. A description of the nonhydrostatic regional COSMO-Model Part I: dynamics and numerics. Deutscher Wetterdienst, Offenbach.
23. Hochreiter, S. and Schmidhuber, J., 1997. Long short-term memory. Neural computation, 9(8), pp.1735-1780.
24. Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H. and Bengio, Y., 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078.
25. Doroshenko, A.Y., Shpyg, V.M. and Kushnirenko, R.V., 2023. Deeplearning-based approach to improving numerical weather forecasts. PROBLEMS IN PROGRAMMING, (3), pp.91-98.
26. Goodfellow, I., Bengio, Y. and Courville, A., 2016. Deep learning. MIT press.
27. https://keras.io/
28. https://www.tensorflow.org/
29. https://www.python.org/

Одержано: 01.12.2023

Про авторів:

Дорошенко Анатолій Юхимович,
доктор фізико-математичних наук,
професор, завідувач відділу ІПС НАНУ,
професор кафедри інформаційних
систем та технологій КПІ
імені Ігоря Сікорського.
Кількість наукових публікацій
в українських виданнях – понад 200.

Кількість наукових публікацій
в зарубіжних виданнях – понад 90.
Індекс Гірша – 7.
http://orcid.org/0000-0002-8435-1451,

Кушніренко Роман Владиславович,
аспірант.
Кількість наукових публікацій
в українських виданнях – 3.
https://orcid.org/0000-0002-1990-8727.

Місце роботи авторів:

Інститут програмних систем
НАН України,
03187, м. Київ-187,
проспект Академіка Глушкова, 40.
Тел.: (38)(044) 526-60-33.
E-mail:
doroshenkoanatoliy2@gmail.com,
roman.kushnirenk@gmail.com.

УДК 004.89 http://doi.org/10.15407/pp2023.04.098

А.Є. Вітюк, А.Ю. Дорошенко

ПРОГРАМНИЙ ПАКЕТ ДЛЯ АДАПТИВНОГО НАВЧАННЯ КОНТРОЛЕРІВ РОБОРУКИ НА ОСНОВІ НЕЙРОМЕРЕЖ

В статті досліджено процес розробки програмних засобів із застосуванням алгоритмів нейроеволюції для навчання нейромережевих контролерів у процесі управління роботизованою кінцівкою. Описано основні аспекти нейроеволюційного підходу для навчання нейромережі у задачах, що не мають найкращого рішення та потребують навчання з підкріпленням. Задача позиціонування має такий ландшафт функцій пристосованості, що робить її схильною потрапляти у пастки локального оптимуму, звідки виникає потреба розробки програмного забезпечення для роботи з нейроеволюційними підходами.

Розробка оптимальних засобів управління із застосуванням нейроеволюційного підходу може забезпечити не лише високу точність позиціонування, а й мінімальну конфігурацію, що підвищує швидкодію системи. Майбутні дослідження з використанням представленого пакету для вирішення задач маніпулювання та позиціонування становлять особливий інтерес для покращення програмних контролерів управління роботизованими кінцівками в динамічних умовах реального світу.

Ключові слова: нейронна мережа, нейроеволюція, роботизована рука, навчання з підкріпленням, NEAT, функція пристосованості.

Вступ

Використання контролерів на основі нейронних мереж у роботизованих системах із робочою кінцівкою-маніпулятором вивела розробку програмного забезпечення для управління роботизованих систем на новий рівень. Використання структур адаптивної нейронної мережі в розробці контролерів для задач маніпулювання рукою робота відкриває нові можливості для дослідження і розробки таких систем. Основна роль адаптивних контролерів випливає з їхньої здатності вирішувати задачі, властиві широкому спектру сценаріїв маніпулювання цільовим об'єктом робочою кінцівкою із захватним пристроєм.

Задача позиціонування роборуки та маніпулювання цільовим об'єктом характеризується своєю складністю, передбачає точний контроль кількох ступенів свободи та взаємодії з динамічним середовищем невідомої конфігурації. Нейронні мережі, з їхньою здатністю навчатися та представляти складні залежності від вхідних даних, пропонують автоматизований шлях для вирішення цих проблем. Крім того, адаптивність структур нейронної мережі додатково надає таким системам здатність динамічно налаштовувати свою архітектуру відповідно до вимог різноманітних завдань, що можуть змінюватися.

Важливість автоматизованої розробки підкреслюється потребою в контролерах, які можуть легко адаптуватися до мінливого середовища, різних форм об'єктів маніпулювання і факторів, пов'язаних із середовищем. Навчальні можливості нейронних мереж, особливо з адаптивними структурами, роблять їх придатними для сценаріїв, де класичне програмування не може забезпечити знаходження мінімальної моделі, розмір якої є критично важливим для забезпечення заданого рівня ефективності.

Процес розробки контролерів на основі нейронних мереж для роборуки має ряд складнощів. Починаючи від отримання та анотації різноманітних наборів даних і закінчуючи складним балансом між складністю моделі та її налаштуваннями. Дослідження еволюційних підходів для навчання адаптивних контролерів дозволить пришвидшити розробку контролерів для широкого спектру можливих сценаріїв використання кінцівки роборуки.

Програмне середовище для тренування агента-маніпулятора з використанням нейроеволюційного підходу має на меті автоматизацію процесу розробки, що є особливо актуальним у сценаріях, де ручне проєктування та налаштування є непрактичним або забирає багато часу.

1. Існуючі програмні рішення

Активний інтерес до дослідження та розробки систем, що базуються на навчанні з підкріпленням, дав поштовх для створення програмних середовищ, які дозволяють легку інтеграцію засобів для навчання моделі та інструментів для роботи з різноманітними агентами та середовищами.

Бібліотека Stable-Baselines3 [1] для навчання з підкріпленням мовою Python забезпечує набір високоякісних реалізацій різних алгоритмів навчання з підкріпленням. Вона створена на основі OpenAI Gym, що дозволяє легко використовувати її для тренувань агентів у різних середовищах, зокрема, для робототехнічних завдань. Бібліотека надає простий і зручний API для роботи з алгоритмами навчання з підкріпленням. Це робить її доступною як для початківців, так і для досвідчених користувачів. Крім того, вона пропонує різноманітні алгоритми навчання з підкріпленням, у тому числі такі популярні, як PPO (наближена оптимізація політики), DDPG (глибокі детерміновані градієнти політики) і TRPO (оптимізація політики довірчого регіону).

Stable-Baselines3 легко інтегрується з OpenAI Gym, набором інструментів для розробки та порівняння алгоритмів навчання з підкріпленням. Це дозволяє користувачам використовувати широкий спектр середовищ, зокрема, для задач маніпуляції з роботизованими робочими кінцівками. Бібліотека забезпечує добре оптимізовану реалізацію різноманітних алгоритмів, що робить її придатною як для досліджень, так і для практичних застосувань.

Проте, хоча Stable-Baselines3 підтримує низку алгоритмів, деякі з них більш придатні для дискретних просторів дій. Для просторів безперервної дії частіше використовуються такі алгоритми, як PPO та TRPO. Крім того, Stable-Baselines3 розроблено для навчання підкріплення з одним агентом. Якщо завдання передбачає мультиагентну взаємодію (наприклад, співпрацю кількох робототехнічних рук для маніпуляції одним об’єктом), інструментів бібліотеки буде недостатньо.

Важливою особливістю є підтримка Stable-Baselines3 фізичного симулятора MuJoCo, що широко використовується для проєктування робототехнічних агентів, зокрема, маніпуляторів та крокуючих роботів.

Альтернативою Stable-Baselines3 є бібліотека TF-Agents з відкритим кодом, побудована на TensorFlow та розроблена Google. Вона забезпечує модульну та розширювану структуру для навчання з підкріпленням. TF-Agents підтримує сценарії як з одним, так і з кількома агентами та має реалізацію різних алгоритмів.

Ray RLlib — ще одна бібліотека для навчання з підкріпленням, створена на основі розподіленої обчислювальної системи Ray. Вона підтримує широкий спектр алгоритмів і пропонує такі функції, як розподілене навчання та налаштування гіперпараметрів.

Для порівняння ефективності навчання контролерів роборуки на основі нейромереж доступні й інші бібліотеки, які мають схожий функціонал, але відрізняються набором алгоритмів навчання з підкріпленням. Серед них Garage, Tianshou, SLM Lab. Порівняння найбільш поширених бібліотек представлено в таблиці 1.

Таблиця 1. Програмні бібліотеки для дослідження методів машинного навчання роботизованих агентів

Бібліотека	Використана бібліотека глибокого навчання	Мультиагентна підтримка	Зручність використання API	Підтримка середовищ OpenAI Gym	Підтримка еволюційних алгоритмів	Переваги
Satble-Baselines3	TensorFlow	Обмежена	Висока	+	-	Великий вибір алгоритмів RL, інтеграція OpenAI Gym

Бібліотека	Використана бібліотека глибокого навчання	Мультиагентна підтримка	Зручність використання API	Підтримка середовищ OpenAI Gym	Підтримка еволюційних алгоритмів	Переваги
TF-Agents	TensorFlow	+	Середня	+	-	Модульність, підтримка сценаріїв з одним або кількома агентами
Ray RLib	TensorFlow, PyTorch	+	Середня	+	+ (Еволюційні стратегії)	Розподілене навчання, налаштування гіперпараметрів, різноманітне API
Garage	TensorFlow	+	Середня	+	+ (Генетичні алгоритми, CEM)	Розширена підтримка алгоритмів
Tianshou	PyTorch	+	Середня	+	+ (CMA-ES)	Модульність, широкий вибір алгоритмів
SLM Lab	TensorFlow	+	Середня	+	+ (CEM)	Модульність

Як бачимо, існуючі програмні рішення надають широкий вибір інструментів для дослідження алгоритмів навчання з підкріпленням для роботизованих агентів у визначеному середовищі. Проте більшість із них не надає підтримки нейроеволюційних підходів, які уможливлюють розробку адаптивних роботизованих систем, що є важливим для дослідження багатофункціональної системи, максимально наближеної до роботи із задачами реального світу.

2. Застосування нейроеволюційного підходу

Генетичні алгоритми — це клас евристик пошуку, натхненних процесом природної еволюції. У природному відборі, ключовому механізмі біологічної еволюції, особини, які є більш придатними для умов, у яких вони живуть, «відбираються» для продовження роду, оскільки вони мають більшу ймовірність вижити до зрілості та знайти пару, ніж менш придатні особини. Генетичні алгоритми застосовують цю концепцію до простору пошуку, де «індивіди» є точками в просторі пошуку, що представляють можливі рішення.

Алгоритм починається з початкового покоління рішення, яке зазвичай генерується випадково. Функція пристосованості оцінює пристосованість кожного кандидата, а алгоритм обирає підмножину кандидатів на основі її значень. Ці кандидати поєднуються один з одним шляхом рекомбінації, щоб сформувати нове покоління рішень-кандидатів з вищою середньою пристосованістю. Залежно від конкретного алгоритму, вони також можуть бути змінені (їхні параметри можуть змінюватися випадковим чином), щоб розширити процес пошуку на нові горизонти простору пошуку. Процес повторюється, доки не буде досягнуто умови зупинки – залежно від алгоритму це може статися через певну кількість поколінь, або коли досягнуто певного рівня фізичної підготовки.

Двома основними елементами потенційного рішення є генотип і фенотип. Генотип — це набір параметрів, які визначають це конкретне рішення. Генотип можна закодувати будь-яким способом за умови, що кодування доступні для пошуку - одним із найпростіших і найпоширеніших кодувань є рядок бітів. Таке кодування називається фенотипом.

Генетичні алгоритми корисні в багатьох галузях, включно із економікою, проєктуванняи систем, криптоаналізом, відеоіграми та логістикою. Основна перевага генетичних алгоритмів перед іншими методами машинного навчання полягає в тому, що вони не

вимагають попереднього аналізу предметної області, оскільки вони починаються з випадкового, неоптимального набору варіантів рішень і використовують еволюційні концепції для пошуку оптимального рішення. Це робить їх дуже корисними для вирішення задач із обмеженими знаннями області. Оскільки мутація гарантує, що їх пошук охоплює різні частини простору рішень, вони також добре справляються з проблемами, де функція помилок має локальні мінімуми.

Нейроеволюція — це еволюційний метод навчання, який поєднує генетичні алгоритми та нейронні мережі, в якому нейронні мережі є фенотипом генетичного алгоритму, а параметри нейронної мережі, такі як топологія або ваги з'єднань, є генотипом [9]. Алгоритм шукає нейронну мережу, оптимальну для певної задачі. По суті, алгоритм розвиває найкращу структуру для даної проблеми.

Алгоритм нейроеволюції може знаходити топологію мережі, або ваги з'єднань, або те й інше. Якщо він шукає обидва параметри, він може розвивати їх окремо один від одного або паралельно. Також може використовуватись пряме кодування, за якого кожен зв'язок і вузол фенотипу вказується в його генотипі, або непряме кодування, коли генотипи просто забезпечують правила для побудови фенотипів.

Нейроеволюційні алгоритми особливо корисні для класів задач, у яких правильна нейронна мережа була б корисною для вирішення проблеми, але важко або неможливо визначити правильно розмічені вхідні дані, необхідні для навчання з учителем. Зазвичай їх визначають як тип навчання з підкріпленням [4].

Поширені алгоритми NEAT і HyperNEAT розвивають як топологію мережі, так і ваги з'єднань. NEAT використовує пряме кодування, а HyperNEAT використовує схему непрямого кодування [2].

3. Алгоритм NEAT для управління агентом із роботизованою кінцівкою

NEAT (NeuroEvolution of Augmenting Topologies) — це алгоритм нейроеволюції, розроблений Остіні 2002 року [3]. Він кодує структуру мережі та ваги з'єднань за допомогою вузлів і використовує біологічні принципи підвищення складності та видоутворення.

Структура генотипу NEAT містить набір генів нейронів і набір генів ланок. Гени нейрона присвоюють нейрону ідентифікаційний номер і вказують, чи є він вхідним, вихідним або прихованим вузлом. Гени зв'язку містять інформацію про два нейрони, до яких вони під'єднані, очікування з'єднання цього зв'язку та прапорці, які вказують, чи є зв'язок активним. Вони також мають номер оновлення, щоб вказати порядок, у якому вони були додані. Щоразу, коли додається з'єднання, алгоритм перевіряє, чи існувало воно раніше (чи воно там було, але видалене, на відміну від того, що воно було повністю новим). Якщо так, йому присвоюється номер попередньої інновації, інакше йому присвоюється наступний доступний.

NEAT розвиває дедалі складніші нейронні мережі відповідно до складності задачі. NEAT розвиває як вагу з'єднання, так і топологію, що приводить до значного підвищення продуктивності. Було показано, що NEAT є ефективним для багатьох задач, таких як балансування маятника, керування роботами, управління транспортними засобами, настільні й відеоігри.

NEAT може формувати різноманітну популяцію мереж шляхом додавання нових генів і розумного схрещування. Однією з проблем цього підходу є те, що менші мережі оптимізуються швидше, ніж великі, а додавання нових зв'язків і генів зазвичай спочатку знижує придатність мережі, тому новостворені структури мають менші шанси вижити. Цю проблему вирішує видоутворення, яке поділяє популяцію на окремі підпопуляції. Структура кожної особини динамічно порівнюється з іншими, а ті, що мають подібну структуру, групуються. Особи в межах виду поділяють загальну пристосованість виду та конкурують переважно всередині цього виду. Специфікація дозволяє оптимізувати нові інновації, не стикаючись з конкуренцією з боку особин із різними структурами.

На рисунку 1 зображено мережу, розроблену для завдання позиціонування кінців-

ки. Мережа є складною та має багато прихованих шарів для ефективної апроксимації нелінійних зворотних кінематичних рівнянь, необхідних для переміщення кінцівки.

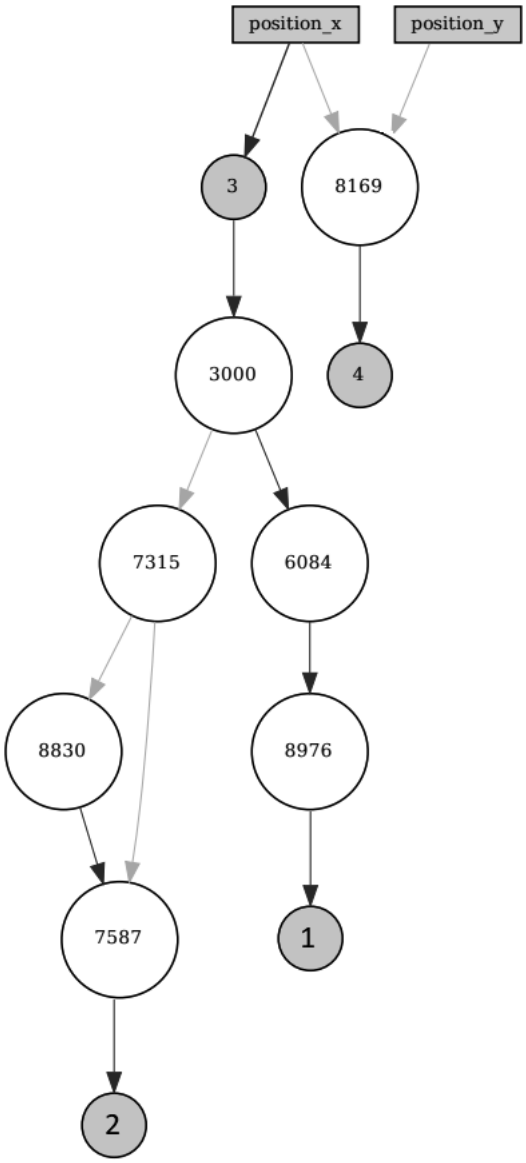

Рис. 1. Мережа для позиціонування роботизованої кінцівки

4. Застосування нейроеволюційного підходу для вхідних даних у форматі зображень

Алгоритм NEAT є ефективним для задачі управління роботизованою рукою, де вхідні дані представлені набором даних з кількох датчиків, наприклад, двома координатами кожного з'єднання. Якщо ж вхідна інформація про стан агента надходить з камери, розміщеної на робочій кінцівці або на основі робота, задача стає на порядок складнішою. Для такої системи доцільним є використання алгоритму HyperNEAT [3].

HyperNEAT (Hypercube-based NeuroEvolution of Augmenting Topologies) є розширенням NEAT, яке використовує форму непрямого кодування під назвою композиційні шаблони створення мереж (CPPN). Воно було створено як спроба скоротити розрив між результатами, отриманими алгоритмами нейроеволюції, та масштабом природного мозку.

У CPPN вхідні дані є декартовими координатами в n-вимірному просторі, де n є

кількістю входів, і передаються у функцію f. Значення f, оцінене за кожною координатою, забезпечує наявність, відсутність або інтенсивність вираження точки в n-вимірному просторі, і тому фенотип, отриманий оцінкою f (який можна розглядати як генотип), є шаблоном. Різниця між звичайними штучними нейронними мережами і CPPN полягає в тому, що штучні нейронні мережі зазвичай містять лише сигмоїдальні функції, тоді як CPPN можуть містити багато різних типів функцій.

Хоча CPPN можна розробити за допомогою генетичних алгоритмів так само, як і штучні нейронні мережі, у HyperNEAT CPPN не розробляються як кінцевий продукт, а використовуються для кодування нейромережі. Оскільки CPPN створює просторовий шаблон, а нейромережа є шаблоном зв’язків, алгоритм потребує відображення між просторовим шаблоном і шаблоном зв’язку. Вхідними даними в CPPN є дві точки, а не одна, і дві точки представляють два вузли в нейромережі, а вихід функції CPPN є вагою зв’язку між двома вузлами. Таким чином, модель, створена CPPN, є топологією нейронної мережі. Цей CPPN, що створює топологію, називається з’єднувальним CPPN, на відміну від просторових CPPN, які створюють просторові моделі. З’єднувальний CPPN бере сітку вузлів, яка називається субстратом, і запитує кожне потенційне з’єднання, для визначення, чи дійсно існує з’єднання та яка його вага, беручи положення двох вузлів, і знаходить вагу з’єднання між ними. Це створює шаблон з’єднань між вузлами в просторі, який є похідним від конфігурації субстрату.

HyperNEAT розвиває CPPN за допомогою стратегії NEAT. CPPN є представленнями шаблонів у гіперпросторі, де кожна точка обмежена чотиривимірним гіперкубом. Отже, шаблон у чотиривимірному гіперкубі можна відобразити на двовимірній нейронній мережі. Першим кроком HyperNEAT є вибір субстрату та зв'язок між входами та виходами. Алгоритм ініціалізує перше покоління мінімальних CPPN з випадковими вагами, так само, як NEAT ініціалізує перше покоління мінімальних штучних нейронних мереж з випадковими вагами. Потім він запитує кожну CPPN, і створює шаблон зв'язку, подібний до нейромережі. Алгоритм визначає придатність цієї нейромережі так само, як це робить NEAT зі своїми мережами, і розвиває друге покоління CPPN відповідно до стратегії NEAT. Це повторюється, як і будь-який алгоритм нейроеволюції, доки не буде знайдено рішення.

HyperNEAT особливо добре відповідає задачам, де важлива інформація про геометричні представлення. За допомогою субстрату він «бачить» координати нейронів. Він може «бачити», який піксель поруч із яким на зображенні, або який квадрат біля якого на шахівниці, замість того, щоб вивчати їх на прикладах, як це робила б традиційна нейронна мережа.

5. Програмна реалізація пакету для адаптивного навчання контролерів робочої кінцівки

Пакет для адаптивного навчання нейромережі, що буде використовуватись для окремої задачі маніпулювання має забезпечувати підтримку як середовища для простору дій агента-роборуки, так і можливості використання необхідного нейроеволюційного алгоритму зі зручним налаштуванням параметрів та інструментами для аналізу результатів.

Для цього розроблений пакет містить набір модулів:

- *Config*: для роботи з параметрами нейроеволюційного алгоритму та його налаштування
- *Population*: для роботи з відповідним типом популяції
- *Reporter*: для забезпечення зручного представлення результатів користувачеві.

Для навчання політики у формі нейромережі для управління робочою кінцівкою в середовищі була обрана бібліотека NEAT-Python як реалізація алгоритму NEAT. На її основі працює модуль *Population.* Бібліотека NEAT-Python забезпечує реалізацію стандартних методів NEAT для моделювання генетичної еволюції геномів організмів у популяції. Вона містить утиліти для перетворення генотипу організму в його фенотип (штучну нейронну мережу)

і надає зручні методи для завантаження та збереження конфігурацій геному разом із параметрами NEAT. Крім того, вона надає корисні процедури, які допомагають збирати статистичні дані про хід еволюційного процесу та зберігати/завантажувати проміжні контрольні точки. Контрольні точки дозволяють нам періодично зберігати стан еволюційного процесу і пізніше відновлювати виконання процесу [1].

Набір середовищ може бути використаний для дослідження завдань роботизованої руки за допомогою наступних наявних інтерфейсів.

Двовимірний симулятор з дискретним простором дій. Робот складається з двох з'єднань, кожне довжиною 100 пікселів, і мета — досягти червоної крапки, яка випадковим чином генерується в кожному епізоді. Простір дій складається з керування кутами з'єднання маніпулятора робота (7 можливих управляючих подій). Простір спостережень дає інформацію про поточний стан навколишнього середовища, він є безперервним і містить відповідну інформацію про цільове та поточне положення з'єднань (4 можливі стани) [7].

Двовимірний симулятор з неперервним простором дій. Подібно до попереднього агента, робот складається з двох з'єднань, кожне довжиною 100 пікселів, і мета — досягти червоної крапки, яка випадково генерується в кожному епізоді. Але простір дій складається з неперервного сигналу керування кутами з'єднання маніпулятора робота (два значення для з'єднань у проміжку -1..1). Простір спостережень є безперервним і містить відповідну інформацію про цільове та потоне положення з'єднань (4 можливі стани).

Тривимірний симулятор PandaReach. Симулятор роборуки Franka Emika Panda, який має на меті позиціонування робочої кінцівки у цільову точку тривимірного середовища. Значення винагороди повертається додатне, якщо цільова позиція досягнута. Управляюча дія відповідає переміщенню кінцевого елемента роборуки.

Тривимірний симулятор KukaCamBulletEnv. Середовище, в якому поточний стан представлений зображенням з камери, що розміщена на платформі-основі робота. Керуючими діями є повороти відповідних кутів з'єднання.

Вибір відповідної модифікації алгоритму NEAT може бути здійснений шляхом заповнення конфігураційного файлу, обробку якого здійснює модуль *Config*:

```
if is_hyper:
   config = GymHyperConfig(args)
if is_eshyper:
   config = GymEsHyperConfig(args)
else:
   config = GymNeatConfig(args)
```

Для прискорення процесу навчання було використано підхід паралелізації обчислень функції пристосованості між процесорами за допомогою класу ParallelEvaluator: neat.ParallelEvaluator(mp.cpu_count(), config.eval_genome, timeout=20)

Розглянемо основний функціонал пакету на прикладі роботи із середовищем двовимірної роборуки з неперервним простором подій.

Параметри обраного середовища та максимальну кількість повторень у підході встановлюються у конфігураційному файлі, який обробляється модулем Config:

```
environment = gym_robot_arm:robot-arm-v1
episode_reps = 10
```

Оптимальний набір параметрів відповідної модифікації алгоритму NEAT також встановлений за допомогою конфігураційного файлу:

```
[NEAT]
compatibility_disjoint_coefficient = 1.1

[DefaultSpeciesSet]
compatibility_threshold = 3.0

[DefaultGenome]
conn_add_prob          = 0.3
conn_delete_prob       = 0.2

[DefaultStagnation]
species_fitness_func = max
max_stagnation       = 20
species_elitism      = 1

[DefaultReproduction]
survival_threshold = 0.2
min_species_size   = 2
```

Після цього обрана функція оцінки може бути встановлена, та запущений процес навчання на кількості поколінь, заданій користувачем:

winner = population.run (parallel_evaluator.evaluate, params.ngen, params.maxtime)

Структура початкової мережі, структура мережі на найкращому геномі та графік видоутворення будуть збережені для подальшого аналізу. Структура мережі-переможця для тестового випадку представлена на рисунку 2:

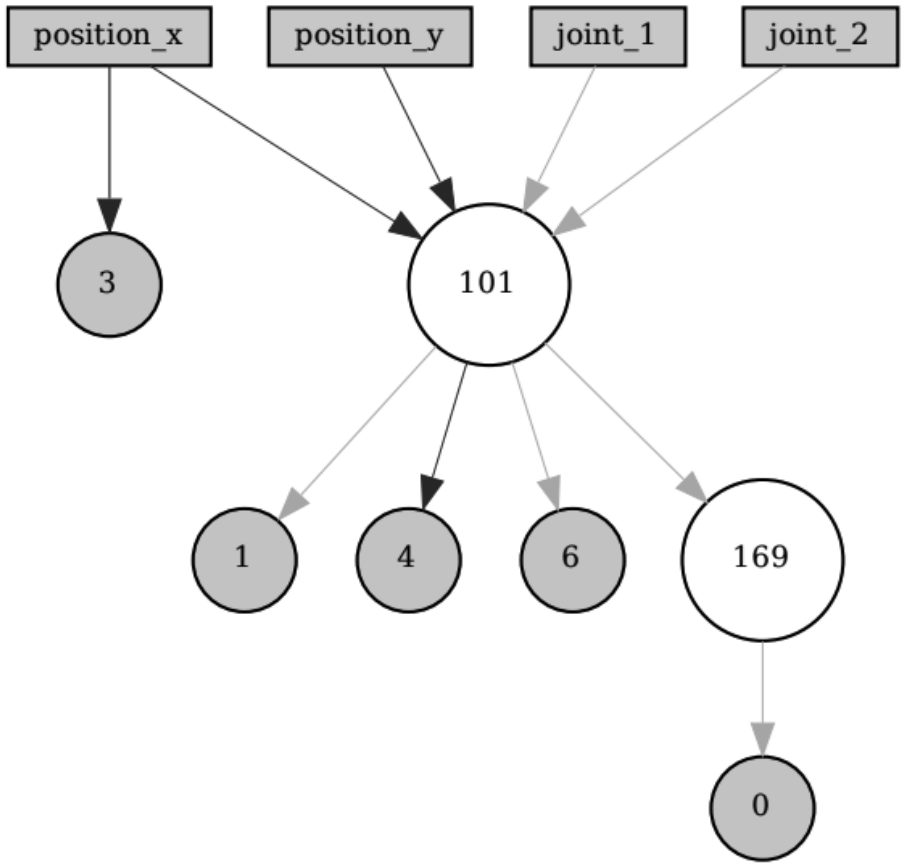

Рис. 2. Структура нейромережі-переможця

Висновки

Розробка програмних засобів для адаптивного навчання управляючих моделей роборуки на основі нейромереж вимагає комплексного підходу, що має враховувати варіативність умов середовища, максимально наближених до реального світу. Позиціонування робочої кінцівки має значний простір пошуку, що може бути також ускладнено різним представленням вхідних даних: як набору значень сенсорів, так і вхідним зображенням. Для умов довільного середовища мають бути використані контролери, що використовують знання, накопичені під час взаємодії з цим середовищем. Інтеграція таких методів штучного інтелекту, як машинне навчання, та навчання з підкріпленням, дала поштовх розвитку програмних засобів для моделювання задач позиціонування рук і маніпулювання об’єктами.

В статті розглянуто основний інструментарій, який має бути представлено у програмному пакеті для ефективного дослідження нейроеволюційного підходу до навчання контролерів роборуки. А також представлено розроблений пакет, який забезпечує доступний інтерфейс для роботи з набором середовищ для виконання задач, які становлять основний інтерес у роботі кінцівки робота.

Залежно від типу вхідних даних та характеристик агента, пакет надає набір інструментів для налаштування як самого агента та його взаємодії з середовищем, так і пареметрів досліджуваного нейроеволюційного алгоритму.

Дослідження конфігурацій NEAT та HyperNEAT покращують здатність алгоритму знаходити інноваційні та ефективні рішення, сприяючи розвитку більш універсальних і потужних архітектур нейронних мереж.

Розробка оптимальних засобів управління із застосуванням нейроеволюційного підходу може забезпечити не лише високу точність позиціонування, а і мінімальну конфігурацію, що підвищує швидкодію системи. Майбутні дослідження з використанням представленого пакету для вирішення задач маніпулювання та позиціонування становлять особливий інтерес для покращення програмних контролерів управління роботизованими кінцівками в динамічних умовах реального світу.

Література

1. Antonin Raffin, et al. Stable-Baselines3: Reliable Reinforcement Learning Implementations, 2021 https://jmlr.org/papers/volume22/20-1364/20-1364.pdf
2. I. Omelianenko, Hands-On Neuroevolution with Python: Build high-performing artificial neural network architectures using neuroevolution-based algorithms, Packt Publishing, 2019
3. Kenneth O. Stanley and Risto Miikkulainen, Evolving Neural Networks Through Augmenting Topologies, Evolutionary Computation 10 (2): 99-127, 2002.
4. Застосування засобів нейроеволюції в технічних системах автоматизації керування / А.Ю. Дорошенко, І.З. Ашур // Проблеми програмування. — 2021. — № 1. — С. 16-25
5. Альона Вітюк, Анатолій Дорошенко. Програмний пакет для оцінки похибка калібрування стереокамери в системі комп'ютерного зору. Проблеми програмування, 2022. № 3-4. С. 469–477. https://doi.org/10.15407/pp2022.03-04.469
6. Neuroevolution of a Robot Arm Controller, 2004. URL: https://www.cs.utexas.edu/ftp/AI-Lab/tech-reports/UT-AI-TR-05-322.pdf
7. OpenAI Gym 2D Robot Arm Environment, URL: https://github.com/ekorudiawan/gym-robot-arm
8. Verbancsics, P., Stanley, K.O., 2010. Evolving static representations for task transfer. Journal of Machine Learning Research. Vol. 11. pp 1737-1769

References

1. Antonin Raffin, et al. Stable-Baselines3: Reliable Reinforcement Learning Implementations, 2021 https://jmlr.org/papers/volume22/20-1364/20-1364.pdf
2. I. Omelianenko, Hands-On Neuroevolution with Python: Build high-performing artificial neural network architectures using neuroevolution-based algorithms, Packt Publishing, 2019
3. Kenneth O. Stanley and Risto Miikkulainen, Evolving Neural Networks Through Augmenting Topologies, Evolutionary Computation 10 (2): 99-127, 2002.
4. Application of neuroevolution tools in technical control automation systems / A.Y. Doroshenko, I.Z. Ashur // Problems of programming. — 2021. — No. 1. — P. 16-25 (in Ukrainian)
5. A. Vitiuk, A. Doroshenko. A software package for estimating stereo camera calibration error in a computer vision system. Problems of programming, 2022. No. 3-4. P. 469–477. https://doi.org/10.15407/pp2022.03-04.469 (in Ukrainian)
6. Neuroevolution of a Robot Arm Controller, 2004. URL: https://www.cs.utexas.edu/ftp/AI-Lab/tech-reports/UT-AI-TR-05-322.pdf
7. OpenAI Gym 2D Robot Arm Environment, URL: https://github.com/ekorudiawan/gym-robot-arm
8. Verbancsics, P., Stanley, K.O., 2010. Evolving static representations for task transfer. Journal of Machine Learning Research. Vol. 11. pp 1737-1769

Одержано: 28.11.2023

***Про авторів*:**

Вітюк Альона Євгеніївна,
аспірантка НТУУ "КПІ імені Ігоря Сікорського".
Кількість наукових публікацій в українських виданнях – 4.

https://orcid.org/0000-0002-1445-9598

Дорошенко Анатолій Юхимович,
доктор фізико-математичних наук,
професор, завідувач відділу теорії комп'ютерних обчислень
Інституту програмних систем
НАН Укрїни,
професор кафедри автоматики і управління в технічних системах НТУУ "КПІ імені Ігоря Сікорського".
Кількість наукових публікацій в українських виданнях – понад 200.
Кількість наукових публікацій в іноземних виданнях – понад 80.
Індекс Гірша – 7.
http://orcid.org/0000-0002-8435-1451,

Місце роботи авторів:
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»,
проспект Перемоги 37.
Інститут програмних систем
НАН України,
03187, м. Київ-187,
проспект Академіка Глушкова, 40.
e-mail: alyonavityuk@gmail.com,
doroshenkoanatoliy2@gmail.com

ПРОБЛЕМИ ПРОГРАМУВАННЯ
2023 – № 4

УДК 004.4

Дослідження ефективності деяких детермінованих методів передобробки сортування даних / В.І. Шинкаренко, О.В. Макаров

Для виконання перевірки гіпотези щодо скорочення часу сортування алгоритмами різної обчислювальної складності здійснені експерименти. Опрацьовано декілька ідей з детермінованої передобробки масивів даних для алгоритмів сортування. Запропоновані відповідні алгоритми: швидка передобробка – передбачення індексу елемента у відсортованому масиві і перестановка, передобробка з пам'яттю – передбачення і перестановка із запам'ятовуванням раніше встановлених елементів, передобробка із розворотом – розвертання послідовностей елементів, відсортованих у зворотному порядку. Також запропоновані блочні варіації швидкої та передобробки з пам'яттю, які виконуються для частин масиву заданої довжини. Встановлено, що більшу ефективність передобробки показують разом із алгоритмами сортування, які значно прискорюються на відсортованих (або майже відсортованих) масивах даних. Блочні передобробки можуть виконуватися швидше через можливість уникнення промахів кешу (cache miss), однак показують нижчий відсоток впорядкованості масиву. Проведені експерименти з оцінки ефективності різних алгоритмів сортування після і разом із запропонованими передобробками.

Ключові слова: алгоритм сортування, сортування, часова ефективність, комбінований алгоритм, передобробка.

UDK 004.4

Study of the efficiency of some deterministic preprocessing methods for sorting algorithms / V.I. Shynkarenko, O.V. Makarov

To verify the hypothesis about decrease in time of sorting by algorithms of different computational complexity experiments have been conducted. Several ideas on deterministic preprocessing of data arrays for sorting algorithms have been tested. The following algorithms are proposed: quick preprocessing – prediction of the index of an element in a sorted array and permutation, preprocessing with memory - prediction and permutation with memorization of previously set elements, preprocessing with reordering – reverting sequences of elements sorted in reverse order. Also proposed block variations of quick and preprocessing with memory, which are performed for parts of the array of a given length. It has been defined that the higher efficiency of preprocessing is achieved by using with sorting algorithms, which are significantly accelerated on sorted (or almost sorted) arrays of data. Block preprocessing methods can be performed faster due to the possibility of avoiding cache misses, but show a lower percentage of array sorting. Experiments were conducted to evaluate the effectiveness of various sorting algorithms after and together with the proposed preprocessing methods.

Keywords: sorting algorithm, sorting, time efficiency, combined algorithm, preprocessing.

УДК 004.65

VuFind: відкрите рішення для інтеграції бібліотечних колекцій / Г.Ю. Проскудіна, К.О. Кудім, В.А. Резніченко.

У статті розглядається система VuFind як відкрите рішення для ефективної інтеграції бібліотечних колекцій. VuFind є потужним пошуковим інтерфейсом, розробленим для поліпшення доступу до різноманітних ресурсів, включно з книжками, статтями, журналами, науковими звітами, іншими матеріалами. Автори обговорюють ключові особливості VuFind, такі як гнучкість налаштувань, можливості пошуку, підтримка метаданих та інтеграція з різними джерелами даних. Наголошується на ролі VuFind у спрощенні пошуку для користувачів та оптимізації управління колекціями з різних бібліотек. VuFind надає відкрите та доступне рішення для створення сучасних бібліотечних систем, сприяючи ефективній інтеграції та підвищенню задоволеності користувачів.

Ключові слова: інтеграція, збір даних, харвестер, протокол OAI-PMH, простий пошук, розширений пошук.

UDC 004.65

VuFind: an open solution for integrating library collections / G.Yu. Proskudina, K.O. Kudim, V.A. Reznichenko.

The article discusses the VuFind system as an open solution for effective integration of library collections. VuFind is a powerful search interface designed to improve access to a variety of resources, including books, articles, journals, scientific reports, and other materials. The authors discuss the key features of VuFind, such as flexible customization, search capabilities, metadata support, and integration with various data sources. They emphasize the role of VuFind in simplifying search for users and optimizing the management of collections from different libraries. VuFind provides an open and available solution for building modern library systems, facilitating effective integration and increasing user satisfaction.

Keywords: integration, data harvesting, harvester, OAI-PMH protocol, simple search, advanced search.

УДК 004.94

Метадані наукових документів як складової системи інформаційних ресурсів «відкритої науки» / О. Захарова

Відкрита наука – це рух, мета якого - зробити результати досліджень, включно з кодом, даними та науковими документами, більш доступними. Він охоплює багато різних, але часто пов'язаних аспектів, що впливають на весь життєвий цикл дослідження, включаючи відкритий доступ до публікацій, відкриті дослідницькі дані, програмне забезпечення з відкритим кодом, відкриті робочі процеси, суспільну науку, відкриті освітні ресурси та альтернативні методи оцінки досліджень, в тому числі відкрите рецензування, експертні оцінки тощо. Тобто створюється наука, заснована на спільних принципах, які забезпечують порівняльність, сумісність і користь для всіх досліджень на всій планеті. Запорукою ефективного застосування та інтеграції ресурсів відкритої науки є їх структурований опис, що базується на принципах повноти та необхідності метаінформації, зручності її використання та інтероперабельності. Одиницями такого опису є метадані.
Дане дослідження не охоплює всього широкого спектру ресурсів відкритої науки. Його метою є структуризація та формалізація лише описів наукових документів як значущої частини наукових знань відкритої науки. Для реалізації поставленої мети в статті визначено таксономію ресурсів відкритих наукових документів та запропоновано єдину інтегровану систему їхніх метаданих, яка містить як множину загальних характеристик, притаманних усім типам наукових документів, незалежно від їхнього призначення, форми чи формату представлення, так і ієрархічно поширювану систему специфічних ознак окремих типів ресурсів.

Ключові слова: ресурси відкритої науки, відкриті наукові знання, метадані наукових документів, схеми метаданих, метадані наукових публікацій, метадані освітніх матеріалів, таксономія понять відкритої науки, інфраструктура відкритої науки, тип відкритого ресурсу, метадані дисертацій, метадані монографій, формати представлення наукових ресурсів.

UDC 004.94

Scientific documents metadata as a component of the system of the "open science" information resources / O. Zakharova

Open science is a movement that aims to make research results more accessible, including code, data, and scientific papers. It covers many different but often related aspects affecting the entire research life cycle, including open access to publications, open data of research, open source software, open workflows, public science, open educational resources and alternative methods of the research evaluation, including open peer review, expert reviews, etc. The key to effective application and integration of open science resources is their structured description based on the principles of completeness and necessity of meta-information, ease of use and interoperability. The units of such description are metadata. In fact, the quality of open resources begins with the quality of its metadata.
This study does not cover the entire wide spectrum of open science resources. Its purposes are to define the system of characteristics that describe general and specific features of various types of scientific documents as a significant part of scientific knowledge of Open Science.
To achieve the goal of the research it is defined a taxonomy of resources of open scientific documents and proposes an integrated system of their metadata. Proposed system of metadata is based on several classifier sets. It includes three major groups: internal characteristics – description of explicit features of the object of open knowledge (for example, the size or the type of the file), administrative characteristics – information about the object (authors, executers, etc.) and descriptive characteristics - information about the object's content, its special features, links to other objects related to this.
The metadata system is built based on the analysis of the existing metadata schemas and standards, search engines and digital libraries, and it takes into account similarity and specificity of each type of open documents.

Key words: open science resources, open scientific knowledge, metadata of scientific documents, metadata schema, metadata of scientific publications, metadata of educational materials, open science concepts taxonomy, open science infrastructure, type of the open resource, metadata of dissertations, metadata of monographs, scientific resources formats.

УДК 681.3

Тривимірна модель семантичного пошуку: запити, ресурси та результати / Рогушина Ю.В.

Запропонована тривимірна модель семантичного пошуку , що аналізує пошукові запити, інформаційні ресурси та результати пошуку, пропонується як додатковий інструмент опису та співставлення інформаційно-пошукових систем (ІПС), що використовують різноманітні елементи штучного інтелекту та менеджменту знань для більш ефективного та пертинентного задоволення інформаційних потреб користувачів. Проаналізовано існуючі підходи до семантизації пошукових запитів та використання зовнішніх джерел знань е процесі виконання пошуку.

Потрібно підкреслити, що значення параметрів, які аналізує ця модель, не є взаємно виключними, тобто та сама ІПС може підтримувати кілька варіантів пошуку. Крім того, засоби подання запитів та ресурсів не завжди є порівнюваними.

Модель дозволяє виявляти ІПС, для яких перетинаються тріади “запит-ІР-результат”, та виконувати їх порівняння саме на цих підкласах пошукових задач. Це дозволяє визначати, які алгоритми пошуку виявляються більш пертинентними для конкретних задач користувачів і на основі цього обирати такі сервіси як джерело інформації для подальшої обробки. Важливою особливістю запропонованої моделі є те, що вона використовує лише ті характеристики ІПС, які можуть бути проаналізовані користувачами.

Ключові слова: семантичний пошук, онтологія, пошуковий запит

UDC 681.3

A three-dimensional model of semantic search: queries, resources, and results / Rogushina J.V.

We propose three-dimensional model of semantic search that analyzes search requests, information resources (IRs) and search results. This model is proposed as an additional tool for describing and comparing information retrieval systems (IRSs) that use various elements of artificial intelligence and knowledge management for more effective and relevant satisfaction of user information needs. In this work we analyze existing approaches to the semanticization of search queries and the use of external knowledge sources for retrieval process.

The values of parameters analyzed by this model are not mutually exclusive, that is, the same IRS can support several search options. More over, the representation means of queries and resources are not always comparable.

The model makes it possible to identify IRSs with intersected triads «request-IR-result» and to perform their comparison precisely on these subclasses of search problems. This approach allows to select search algorithms that are more pertinent for specific user tasks and to choose on base of this selection appropriate retrieval services that provide information for further processing. An important feature of the proposed model is that it uses only those IRS characteristics that can be directly evaluated by retrieval users.

Keywords: semantic search, ontology, search query

UDC 517.958:57 +519.711.3 + 612.51.001

Simulating of human physiological super-systems: modeling of kidney and bladder functions / R.D Grygoryan., A.G.Degoda, T.V.Lyudovyk, O.I. Yurchak

A special quantitative model describing functions of human kidney and bladder is created. Currently, the model is realized and tested as an autonomous C# software module (SM) functioning under given dynamic input characteristics. Later, SM will be incorporated into our specialized software capable to simulate the main modes of human integrative physiology namely, interactions of physiological super-system (PSS). The model of kidney describes mechanisms of blood filtration in Bowman's capsule, reabsorption in collecting tubules, as well as the mechanism of central renin-angiotensin system. The model of bladder describes the dynamics of its filling and periodic emptying. Each act of bladder emptying is initiated by a signal generated by brain in response to afferent impulses pattern from bladder's mechanoreceptors. Models have been tested using algorithms providing designing of scenarios including simulation of either short-time or long-time (hours or days) observations. Input data include different combinations of pressure in renal afferent arterioles, blood osmotic, and oncotic pressures. Output data includes dynamics of primary urine, final urine, bladder volume, urine pressure, mechanoreceptors' activity, renin production velocity, blood renin concentration, angiotensin2 production velocity, and blood angiotensin2 concentration, as well as blood albumin and sodium concentrations. Both student-medics and physiologists interested in providing of theoretical research can be users of SM.

Key words: physical health, kidney, bladder, physiological mechanisms, quantitative model, simulator.

УДК 517.958:57 +519.711.3 + 612.51.001

Симулятор фізіологічних надсистем людини: моделювання функцій нирок та січового міхура / Р.Д.Григорян, А.Г.Дегода, Т.В.Людовик, О.І.Юрчак

Створено кількісну модель, що описує функції нирок і сечового міхура людини. Модель реалізовано та протестовано як автономний програмний модуль C# (SM), що функціонує при заданих динамічних вхідних характеристиках. Пізніше SM буде включено в наше спеціалізоване програмне забезпечення, що буде здатне моделювати основні режими інтегративної фізіології людини, а саме взаємодії фізіологічної суперсистеми (PSS). Модель нирок описує механізми фільтрації крові в капсулі Боумена, реабсорбції в збиральних канальцях, а також механізм центральної ренін-ангіотензинової системи. Модель сечового міхура описує динаміку його наповнення та періодичного випорожнення. Кожен акт випорожнення сечового міхура ініціюється сигналом, який генерується мозком у відповідь на аферентні імпульси від механорецепторів сечового міхура. Моделі були перевірені з використанням алгоритмів, які розробляють сценарії, в тому числі симуляцію короткочасних або тривалих (години або дні) спостережень. Вхідні дані включають різні комбінації тиску крові в ниркових аферентних артеріолах, осмотичний і онкотичний тиски крові. Вихідні дані включають динаміку первинної сечі, кінцевої сечі, об'єму сечового міхура, тиску сечі, активності механорецепторів, швидкості продукції реніну, концентрації реніну в крові, швидкості продукції ангіотензину2, концентрації ангіотензину2 в крові, а також концентрації альбуміну та натрію в крові. Користувачами СМ можуть бути як студенти-медики, так і фізіологи, зацікавлені у проведенні теоретичних досліджень.

Ключові слова: фізичне здоров'я, нирка, сечовий міхур, фізіологічні механізми, кількісна модель, симулятор.

УДК 519.85

Інсерційна семантика квантових взаємодій / Ю.Г. Тарасіч, Г.А. Солошенко

У статті розглядається застосування методу інсерційного моделювання, а саме - теорії агентів та середовищ на основі алгебри поведінок, до моделювання експериментів, спрямованих на вивчення фізичних, хімічних та біологічних процесів.
Даний підхід використовує багаторівневу модель взаємодій, в основі якої лежать знання про квантові взаємодії в середовищі атомів та молекул. Усі інші види взаємодій (міжмолекулярні взаємодії, хімічні реакції, взаємодії біологічних об'єктів тощо) розглядаються та моделюються як наслідки формалізованих законів квантової взаємодії. Застосування алгебраїчного підходу дозволяє розглядати як експерименти, що стосуються вивчення властивостей певного процесу та визначення його кінцевого результату в термінах середовища експерименту, так і можливість обереного моделювання, коли задано «кінцевий результат» (властивості певної фізичної сутності – речовини або процесу) та необхідно знайти початкові параметри і відповідні дії, що приводять до нього.
У статті наведено приклади формального представлення основних агентів, що розглядаються у моделях, їхні взаємодії, а також дослідження властивостей речовини на прикладі виявлення речовини з магнітоелектричними властивостями.

Ключові слова: інсерційне моделювання, квантові взаємодії, молекулярне моделювання, алгебраїчне моделювання.

UDC 519.85

Insertion semantics of quantum interactions / Yu.G.Tarasich, H.A. Soloshenko

The rapid development of the chemical industry and science and new challenges in the field of health care put forward increased demands for the development of the theory of organic and inorganic chemistry, biochemistry and biophysics, the search and implementation of new modelling and analysis methods, and the improvement of technological processes.
One of the safe and fast methods of researching the properties and behavior of new materials and tools is the modelling of relevant experiments, in particular, computer molecular modelling based on mathematical models.
Modelling the interactions between micro and macromolecules at the quantum level allows us to manipulate the substances' electronic, magnetic, optical and other characteristics and consider the possibilities of creating new chemical bonds, molecular structures, phase transitions, quantum states, and so on.
Accordingly, the main idea of our research is to apply the technology of algebraic modelling and quantum-chemical apparatus for the simulation and verification of experiments in physics, chemistry, and biology areas.
The use of formal algebraic methods allows proving properties and finding relevant scenarios for the effective analysis of the behavior of various objects in real-time, considering not individual scenarios but sets of possible behaviors.
At this research stage, we have developed a methodology for formalization complex organic and inorganic substances, chemical processes and reactions based on the formalization of the interaction of atoms and molecules at the level of quantum interactions.

Keywords: insertion modelling, quantum interactions, molecular modelling, algebraic modelling.

UDC 004.8

Simulation of the autonomous maze navigation using the NEAT algorithm / Ia.V.Omelianenko

The article deals with the problem of finding a solution for the navigational task of navigating a maze by an autonomous agent controlled by an artificial neural network (ANN). A solution to this problem was proposed by training the controlling ANN using the method of neuroevolution of augmenting topologies (NEAT).
A description of the mathematical apparatus for determining the goal-oriented objective function to measure fitness of the decision-making agent, suitable for optimizing the training of ANN in the process of neuroevolution, was given. Based on the invented objective function, a software was developed to control the neuroevolutionary process using the Python programming language.
A system for simulating the behavior of an autonomous robot that can navigate through a maze using input signals from various types of sensors has been created. The simulation system allows to imitate the behavior of a physical robot in a large number of experiments in a short time and with minimal expenses.
The experiments performed using the created simulation system to find the optimal values of hyperparameters, which can be used for successful training of the controlling ANN by the method of neuroevolution, are presented.
Additionally, the implemented new methods of visualizing the training process are described. These methods significantly simplify the search for optimal hyperparameters of the NEAT algorithm, due to the visual demonstration of the effect of changing one or another parameter on the training process.

Keywords: genetic algorithms, neuroevolution of augmenting topologies, autonomous maze navigation, NEAT, autonomous robot simulator.

УДК 004.8

Моделювання автономного проходження лабіринту з використанням алгоритму NEAT / Я.В.Омельяненко

У статті розглянуто проблему пошуку рішення для навігаційної задачі проходження лабіринту автономним агентом під керуванням штучної нейронної мережі (ШНМ). Було запропоноване рішення цієї задачі за рахунок навчання контролюючої ШНМ з використанням методу нейроеволюції наростаючої топології (NEAT).
Було наведено опис математичного апарату для визначення цільової функції пристосованості агента-вирішувача, придатної для оптимізації процесу навчання в процесі нейроеволюції. На основі винайденої цільової функції було розроблене програмне забезпечення для керування нейроеволюційним процесом із використанням мови програмування Python.
Окрім цього, було створено систему моделювання поведінки автономного робота, здатного пройти оточення лабіринту використовуючи вхідні сигнали від датчиків різних типів. Створена система моделювання дозволяє імітувати поведінку фізичного робота у великій кількості експериментів у стислі терміни та з мінімальними витратами.
Надано опис експериментів виконаних з використанням створеної моделюючої системи для пошуку оптимальних величин гіперпараметрів, що можуть бути використані для успішного навчання керуючої ШНМ методом нейроеволюції.
Надано опис розроблених нових методів візуалізації процесу навчання агентів-вирішувачів. Розроблені методи значно полегшують пошук оптимальних гіперпараметрів алгоритму NEAT за рахунок наочного представлення впливу зміни того чи іншого параметру на процес навчання.

Ключові слова: генетичні алгоритми, нейроеволюція наростаючих топологій, автономне проходження лабіринту, NEAT, симулятор автономного робота.

УДК 51:681.3.06

Рекурентні нейронні мережі для задачі уточнення чисельних метеорологічних прогнозів / А.Ю. Дорошенко, Р.В. Кушніренко

Зроблено короткий огляд прикладів застосування “глибокого навчання” до геонаукових задач. Описано основні проблеми, що виникають у застосуванні “глибокого навчання” до задач метеорологічного прогнозування. Порівняні два найпопулярніші види архітектур рекурентних нейронних мереж, а саме мережу довгої короткочасної пам’яті та вентильний рекурентний вузол, у застосуванні до постпроцесингу результатів прогнозу приземної температури, отриманого за допомогою чисельних гідродинамічних методів метеорологічного прогнозування. Порівняння ефективностей двох архітектур рекурентних нейронних мереж було здійснене за допомогою середнього квадратичного відхилення. Показано, що усі моделі з вентильними рекурентними вузлами є ефективнішими за моделі довгої короткочасної пам’яті. Виявлено найкращу архітектуру рекурентних нейронних мереж для розв’язання задачі уточнення чисельних метеорологічних прогнозів.

Ключові слова: “глибоке навчання”, рекурентні нейронні мережі, метеорологічне прогнозування.

UDC 51:681.3.06

Recurrent neural networks for the problem of improving numerical meteorological forecasts / A.Yu. Doroshenko, R.V. Kushnirenko

This paper briefly describes examples of how deep learning can be applied to geoscientific problems, as well as the main difficulties that arise when scientists apply this technique to the problems of meteorological forecasting. This paper aims at comparing the two most popular types of recurrent neural network architectures, namely the long short-term memory network and the gated recurrent unit when they are used to improve 2m temperature forecast results obtained using numerical hydrodynamic methods of meteorological forecasting. An efficiency comparison of architectures of recurrent neural networks was performed using the root-mean-square error. It is shown that all models with gated recurrent units are more efficient than models with long short-term memory. Thus the best architecture of recurrent neural networks for solving the problem of improving numerical meteorological forecasts has been revealed.

Key words: deep learning, recurrent neural networks, meteorological forecasting.

UDC 004.89

Software package for adaptive training of robot controllers based on neural networks / A.Y. Vitiuk, A.Y. Doroshenko

The article deals with the development of a software solution for the application of neuroevolution algorithms when creating a controller for controlling a robotic arm. The main principles of the neuroevolutionary approach for training neural network controllers in tasks requiring reinforcement learning are considered. In particular, the advantages of the adaptive approach are determined for a wide class of scenarios in which the working limb can work: implementation of stable grasping, positioning, manipulation of objects. It is noted that the final result of neuroevolution is an optimal network topology, which makes the model more resource-efficient and easier to analyze. The paper considers a software system that provides the developer with all the necessary tools for modeling the behavior of a robotic agent in environments of various levels of complexity: both two-dimensional and three-dimensional. In addition, the possibility of specifying the state of the agent not only as a set of data from sensors, but also as an image of the current environment from the camera is considered. According to the results of the experiments, the high efficiency of the search for the best solution using the NEAT algorithm is noted. It has been established that the proposed solution allows productively obtaining an effective policy in the form of a neural network, which has a minimal configuration, which will allow to increase the speed of the controller, which is critical for the operation of a real system.
Thus, the use of a software solution for the adaptive development of a neuroevolutionary controller for solving tasks with a robotic limb allows to increase the efficiency of the learning process and obtain an optimal network topology.

Keywords: neural network, neuroevolution, robotic arm, reinforcement learning, NEAT, fitness function.

УДК 004.89

Програмний пакет для адаптивного навчання контролерів роборуки на основі нейромереж / А.Є. Вітюк, А.Ю. Дорошенко

У статті розглядається розробка програмного рішення для застосування алгоритмів нейроеволюції в процесі створення контролера для управління роботизованою рукою. Розглянуто основні принципи нейроеволюційного підходу для навчання нейромережевих контролерів у задачах, що потребують навчання з підкріпленням. Зокрема, визначено переваги адаптивного підходу для широкого класу сценаріїв, в яких може працювати робоча кінцівка: здійснення стійкого захвату, позиціонування, маніпулювання об’єктами. Відмічається, що кінцевим результатом нейроеволюції є оптимальна топологія мережі, яка робить модель більш ресурсоефективною та легшою для аналізу. В роботі розглядається програмна система, яка надає розробнику всі необхідні інструменти для моделювання поведінки агента-роборуки у середовищах різного рівня складності: як двовимірних, так і тривимірних. Крім того, розглянуто можливість задання стану агента не лише як набору даних з датчиків, а і як зображення поточного середовища з камери. Згідно результатів експериментів відмічається висока ефективність пошуку найкращого рішення із застосуванням алгоритму NEAT. Встановлено, що запропоноване рішення дозволяє продуктивно отримати ефективну політику в формі нейромережі, яка має мінімальну конфігурацію. Це дозволить підвищити швидкодію контролера, яка є критичною для роботи реальної системи.
Таким чином, використання програмного рішення для адаптивної розробки нейроеволюційного контролеру для вирішення задач роботизованою кінцівкою дозволяє підвищити ефективність навчального процесу та отримати оптимальну топологію мережі.

Ключові слова: нейронна мережа, нейроеволюція, роботизована рука, навчання з підкріпленням, NEAT, функція пристосованості.

ДО УВАГИ АВТОРІВ!

У журналі «Проблеми програмування» публікуються наукові матеріали, які раніше не публікувалися в інших виданнях.

Мова статті: українська, англійська.* Обсяг статті - від 6 до16 сторінок формату А4.

Документ зберігається у форматі doc або docx. Ім'я подається транслітерацією, як прізвище автора (авторів), наприклад, "Petrenko.doc".

Автори можуть користуватися електронною поштою і також телефаксом для ділової переписки та передачі до редакції тексту статті та правки при коректурі. E-mail редакції: alengoro@isofts.kiev.ua. FAX: +380 (44) 526 6263, Телефон: +380 (96) 418 3082.

1. Оформлення файлу з текстом статті.

При підготовці файлу використовуються: стиль нормальний (звичайний) або normal; шрифт Times New Roman, розмір шрифта 12 пт.; міжрядковий інтервал – 1,0; абзацний відступ – 1,25 см; вирівнювання – по ширині. У тексті не допускається вирівнювання пропусками; розстановка переносів – автоматична. Формат паперу А4, розміри полів документа – 20 мм. Текст статті після анотації має бути оформлений **у 2 колонки,** ширина яких – 7,86 см, а пробіл між ними – 1,27 см.

2. Послідовність розміщення та оформлення матеріалу статті.

УДК: індекс за універсальною десятковою класифікацією.

Автори: ініціали та прізвища авторів, курсив (світлий).

Заголовок 1 **(назва статті):** не містить абревіатур та строго відповідає змісту статті. Шрифт 15 пт, напівжирний, регістр верхній.

Анотація **(мовою статті):** 50-100 слів, не містить абревіатур, зрозумілих із змісту статті. Шрифт 10 пт, звичайний.

Ключові слова **(мовою статті):** не більше10 слів, не містить абревіатур, зрозумілих із змісту статті, подаються в називному відмінку, розділені комами. Шрифт 10 пт, звичайний.

Заголовок 2 **(назва розділу):** шрифт 14 пт, напівжирний; абзац із центральним вирівнюванням, без переносів. Заголовки нижчого рівня (пункти і т.п.) у самостійний абзац не виділяються і проходять першим реченням текстового абзацу, шрифт 12 пт, напівжирний.

Основний текст статті має такі необхідні елементи:

постановка проблеми в загальному вигляді і її зв'язок з важливими науковими або практичними завданнями;

аналіз останніх досліджень і публікацій, у яких розпочато рішення даної проблеми і на які спирається автор, виділення невирішених раніше частин загальної проблеми, яким присвячується дана стаття;

формулювання цілей статті (постановка задачі);

виклад основного матеріалу дослідження з повним обґрунтуванням отриманих наукових результатів;

висновки з даного дослідження і перспективи подальших розробок у даному напрямку;

подяка (за наявності такої).

Формули створюються в редакторі Microsoft Equation 3.0 або MathType. Формули, на які є посилання в тексті, повинні мати наскрізну нумерацію. Номер формули друкується в круглих дужках біля краю правого поля. Розмір основного шрифту редактора формул – 12 пт. Розміри символів у формулах: звичайний – 12 пт, великий індекс – 9 пт, дрібний індекс – 7 пт, великий символ – 18 пт, дрібний символ – 11пт. Не допускається масштабування формульних об´єктів.

Рисунки мають бути створені вбудованим редактором Word Picture або експортовані з прикладних програм Windows у графічних форматах (bmp, pcx, gif, jpg або tif). Рисунки розташовуються по центру. Нумерація рисунків здійснюється відповідно до порядку

згадування у тексті. Нумеровані підписи розміщуються під рисунком з позначенням «Рис. », далі вказується номер рисунка і текст підпису.

Таблиці мають бути підготовлені стандартним вбудованим в Word інструментарієм "Таблиця". Таблиці нумеруються за порядком згадування. На номер таблиці повинно бути посилання в тексті. Номер таблиці вказується в окремому рядку з вирівнюванням по правій стороні (наприклад, «Таблиця 1»). Назви таблиць розміщуються над таблицею з вирівнюванням по центру. Мінімальний розмір шрифту в таблицях – 11 пт.

Література**:** нумерований список джерел згідно ДСТУ 8302:2015 від 01.07.2016 р., шрифт 11 пт, відступ: спеціальний, навислий, 0,63 см. Джерела з заголовками на латиниці наводяться без перекладу. Інші джерела подаються мовою оригіналу. Приклади оформлення бібліографічних посилань згідно з вимогами ***Harvard Style*** наведені в багатьох публікаціях, наприклад: http://www.staffs.ac.uk/assets/harvard_referencing_examples_tcm44-39847.pdf

Дані про авторів**:** мають починатися рядком "Про авторів:", напівжирний курсив. Далі вказуються для кожного з авторів ПІБ повністю, наукове звання, посада, адреса, кількість публікацій в українських виданнях (приблизна), кількість публікацій в зарубіжних індексованих виданнях (приблизна), індекс Хірша (за наявності), обов'язково номер ORCID (сайт ORCID http://orcid.org/).

Дані про місце роботи авторів**:** починаються рядком "Місце роботи авторів:", напівжирний курсив. Далі вказуються місце роботи, адреса, телефон, факс, електронна пошта, контактний телефон.

3. Оформлення файлу з анотаціями.

Файл з анотаціями містить інформацію двома мовами (наприклад, якщо стаття написана українською мовою, то анотація та ключові слова – англійською і українською мовами) та має бути оформлений у дві колонки: УДК (шрифт – 8 пт); назва статті (шрифт – 12 пт, напівжирний); прізвища та ініціали авторів (шрифт – 12 пт); текст анотації, ключові слова (шрифт – 10 пт).

Вимоги до анотації англійською мовою: обсяг від 100 до 250 слів, інформативність, оригінальність (не є калькою української анотації), змістовність (відображає основний зміст статті і результати досліджень), структурованість (дотримується логіки опису результатів у статті).

Документ зберігається у форматі doc або docx. Ім´я подається транслітерацією, як прізвище автора (авторів), наприклад, "Petrenko_Annot.doc".

*16.07.2020 р. набули чинності положення Закону України «Про забезпечення функціонування української мови як державної». Відповідно до статті 22 «Державна мова у сфері науки» у наукових виданнях не повинно бути вміщено матеріалів іншими мовами, окрім державної, англійської та мов ЄС.

Примітка: Підписний індекс журналу «Проблеми програмування» – **90853**.