УДК 004.021: 81’32
DOI https://doi.org/10.32838/2663-5941/2020.1-1/15

Кузьма К.Т.
Миколаївський національний університет імені В.О. Сухомлинського

Мельник О.В.
Миколаївський національний університет імені В.О. Сухомлинського

ОБЧИСЛЮВАЛЬНА ТЕХНОЛОГІЯ ПЕРЕВІРКИ ВІДПОВІДЕЙ У СИСТЕМАХ ТЕСТУВАННЯ

Актуальність дослідження зумовлена бурхливим розвитком дистанційної освіти, інтелектуалізацією процесу перевірки відповідей у системах тестування. За результатами дослідження методів порівняння рядків визначено можливість їх застосування в системах тестування для встановлення рівня відповідності короткої текстової відповіді еталону (правильній відповіді).

Запропоновано обчислювальну технологію (ОТ) перевірки короткої відповіді в системах тестування, наданої у вигляді тексту. ОТ містить чотири базових етапи: нормалізація відповіді та еталону (приведення в нижній регістр, видалення всіх символів, відмінних від літер, цифр і пробілу); виділення слів (із речень видаляються прийменники, сполучники); ітеративне порівняння слів відповіді та еталону шляхом обчислення найбільшої загальної підпослідовності (LCS), використовуючи модифікацію методу динамічного програмування, алгоритм Хешберга; знаходження показника збігу для всього речення шляхом додавання LCS окремих слів. Коефіцієнт збігу відповіді з еталонною визначається під час тестування алгоритму. Такий підхід передбачає, що чим більшою є довжина найбільшої загальної підпослідовності, тим більшою є відповідність правильній відповіді.

Методи динамічного програмування використано для зіставлення коротких відповідей, які містять неточності формулювання із правильною відповіддю, що дозволило вирішити задачу перевірки відповіді, поданої природною мовою.

Розроблено обчислювальну технологію перевірки відповіді, поданої у довільній текстовій формі, для автоматизації процесу тестування в системах типу «запитання – коротка відкрита відповідь». Подальші дослідження будуть зосереджені на алгоритмах «приблизного» порівняння рядків, які належать до задач рядкового «нечіткого» пошуку.

Ключові слова: *задача «нечіткого» пошуку, алгоритм Хешберга, методи порівняння рядків, системи тестування, методи динамічного програмування, найбільша загальна підпослідовність.*

Постановка проблеми. В існуючих системах автоматизованої перевірки знань обмежено функціональність застосування неформалізованого підходу побудови тестових завдань. Додатки, які використовуються для перевірки рівня знань та умінь здобувачів вищої освіти (Moodle, OpenTEST), підтримують побудову тестових запитань певних типів, наприклад «оноваріанті: одне запитання – один вірний варіант відповіді», «множинного вибору: одне запитання – дві та більше вірних відповіді», «зіставлення визначень і відповідей», «визначення правильної послідовності дій, етапів, процесів» тощо. При цьому питання типу «запитання – коротка відповідь» вимагають повного збігу відповіді з еталоном/еталонами, який/які зберігаються в базі даних. Таким чином, для підвищення ефективності процесу автоматизованої перевірки відповіді, поданої природною мовою, актуальними є дослідження технології застосування методів, алгоритмів «нечіткого» порівняння рядків в автоматизованих системах тестування.

Аналіз останніх досліджень і публікацій. Механізми застосування природної мови, її формалізації в автоматизованих системах тестування, перевірки рівня знань набувають актуальності у зв’язку із розвитком дистанційної освіти, цифровізації та інтелектуалізації навчального процесу.

Розглядаючи алгоритми порівняння рядків, було здійснено їх класифікацію на два основних види: «точного» порівняння із зразком (паттерном), «приблизного, нечіткого» порівняння з «паттерном». При цьому шаблон (паттерн) пошуку може бути одиночним або множинним [1–2].

Алгоритми «нечіткого» пошуку характеризуються метрикою – функцією відстані між двома словами, яка дозволяє оцінити ступінь їх подібності. Найбільш відомі метрики: відстань Хеммінга, Левенштейна та Дамерау-Левенштейна. При цьому відстань Хеммінга є метрикою тільки

на безлічі слів однакової довжини, що обмежує область її застосування.

Моделі та методи «нечіткого» пошуку досліджувалися науковцями О.М. Лесько, О.І. Комарницькою, О.В. Палагіним, І.С. Катеринчук, А.Ю. Гулою, А.П. Ігнатенком, А.В. Чадюком та іншими [3–9]. Наведені в роботах [3–7] методи в основному базуються на модифікації метрики Левенштейна, коли є мінімальна кількість операцій (вставки, видалення або заміни символу), необхідних для перетворення одного рядка в інший (редакційна відстань).

Іншим підходом до визначення подібності рядків є застосування фонетичного кодування. Фонетичні алгоритми співставляють двом словам зі схожою вимовою однакових кодів, що дозволяє здійснювати порівняння та індексацію безлічі таких слів на основі їх фонетичної подібності: Soundex, Daitch-Mokotoff Soundex, NYSIIS, Metaphone, Double Metaphone, Caverphone [10]. Такі алгоритми зручно використовувати при пошуку в базах за списками людей, у програмах перевірки орфографії. Найчастіше вони використовуються спільно з алгоритмами «нечіткого» пошуку, надаючи користувачам зручний пошук за іменами та прізвищами в різних базах даних. Більшість фонетичних алгоритмів є мовнозалежними. Як правило, вибір рішення в кожному конкретному випадку залежить від особливостей предметної області та постановки завдання.

Постановка завдання. Мета роботи – розробка обчислювальної технології зіставлення відповідей, поданих у довільній текстовій формі, з правильними відповідями, які зберігаються в базі даних, для автоматизації процесу тестування в системах типу «запитання – коротка відкрита відповідь».

Виклад основного матеріалу дослідження. Формалізація природної мови в автоматизованих інформаційних системах базується на математичних методах: теорія графів, комбінаториці, нечітких множинах; теорії ймовірності та математичної статистики, методах штучного інтелекту.

Запропонована обчислювальна технологія призначена для порівняння тестових відповідей на запитання в системах тестування з правильними відповідями, заданими невеликими реченнями (до 10 слів), шляхом обчислення міри, яку називають найбільшою загальною підпослідовністю двох слів (LCS – Longest Common Subsequence).

Найбільшою загальною підпослідовністю (longest common subsequence, LCS) слів x та y буде слово найбільшої довжини, яке одночасно є підпослідовністю x і x. Довжина LCS тісно пов'язана з редакційною відстанню. Вона є зворотною відстані Левенштейна: чим більше збігається підпослідовність, тим менше операцій необхідно для перетворення.

Обчислення довжини найбільшої загальної підпослідовності двох слів здійснено з використанням алгоритму Хешберга [11; 12, с. 287]. Алгоритм є рекурсійним, на кожній ітерації обчислюються значення довжин найбільших спільних підпослідовностей слів, які записуються до масиву L. Автори позначають їх як $L[i, j]$:

$$L[i, j] = \left| LCS\left(x[1..i], y[1..j]\right)\right|.$$

Функція LCS (x, y) визначає найбільшу загальну підпослідовність слів x та y, де x – слово із наданої відповіді, y – слово із правильної відповіді, яка зберігається в базі даних. Оскільки довжина LCS будь-якого слова зі словом, яке не містить жодного символу, дорівнює нулю, значення границь масиву L встановлюються як $L[i,0] = L[0,j] = 0$. Елементи масиву L із індексами $[i, j]$ визначаються в процесі порівняння елементів $x[1, i]$ та $y[1, j]$ за такими правилами:

1) якщо $x_i = y_j$, додаємо одиницю до поточного значення $L[i, j]$ префіксів $x[1, i-1]$ та $y[1, j-1]$. Таким чином $L[i, j] = L(i-1, j-1) + 1$;

2) якщо $x_i \uparrow y_j$, значення $L[i, j]$ обчислюється як максимум із попередніх сусідніх значень: $L[i, j] = \max\{L[i-1, j], L[i, j-1]\}$ *:*

$$\begin{cases} L[i,0] = 0, 0 \le i \le n; \\ L[0,j] = 0, 0 \le j \le m; \\ L[i,j] = \begin{cases} L(i-1, j-1) + 1, \textit{ якщо } x_i = y_j; \\ \max\{L[i-1, j], L[i, j-1], \textit{ в іншому випадку }/ \end{cases} \end{cases}$$

Значення довжини LCS слів x та y буде максимальним значенням масиву L, яке знаходиться в комірці $L[n, m]$. Витрати алгоритму щодо пам'яті та часу обчислення складають $Q(n \times m)$, де n і m – довжини рядків, які порівнюються.

Одним із варіантів спрощення алгоритму, який дозволяє зменшити вимоги до необхідного об'єму пам'яті до величини $Q\left(\min\{n, m\}\right)$, є введення додаткової змінної q, за допомогою якої здійснюється переадресація між двома одновимірними масивами розмірністю $\min(n, m)$. Для цього необхідно задати масив $Ls[0..1, 0..m]$ розмірністю $2 \times m$, у якому зберігався б поточний - i та попередній - $i-1$ рядки масиву L. З використанням змінної q $(q \in 0..1)$ здійснюється доступ до

елемента $Ls[q, j]$ поточного рядка, тоді як значення $1 - q$ визначають елементи $Ls[1 - q, j]$ попереднього. Спрощений алгоритм повертає останній рядок $L[n, 0..m]$ масиву L, максимальне значення в якому визначає LCS. Лістинг коду методу для визначення LCS на мові C#:

```
static int FindLS (string x, string y);
{int n = x. length;
int m = y. length;
int i, j;
int [,] Ls = new int [2, m + 1];
for (j = 0; j < = m; j + +);
Ls [1, j] = 0;
For (i = 1; i < = n; i + +);
{for (j = 0; j < = m; j + +);
Ls [0, j] = Ls [1, j];
For (j = 1; j < = m; j + +);
{if (x [i-1] = y [j-1]);
Ls [1, j] = Ls [0, j-1] + 1;
Else
Ls [1, j] = max (Ls [1, j-1], Ls [0, j]); // max () –
функція // визначення максимального з двох
чисел
};
}return Ls [1, m];
}.
```

Запропонована обчислювальна технологія перевірки відповіді в системах тестування, поданої у текстовій формі, передбачає виконання таких етапів:

1) Нормалізація наданої відповіді та еталону. Рядки переводяться в нижній регістр, видаляються всі символи, відмінні від букв, цифр і пробілу.

2) Виділення слів. Словами вважаються всі послідовності символів без пробілів, які мають довжину > = 3. Цим самим видаляються майже всі прийменники, сполучники тощо.

3) Порівняння слів шляхом обчислення LCS, використовуючи алгоритм динамічного програмування.

4) Знаходження загального показника подібності для всього речення шляхом додавання LCS усіх слів.

Такий підхід передбачає, що чим більшою є довжина LCS, тим більшою є відповідність правильній відповіді. Показник подібності відповіді, наданої природною мовою з еталоном, підбирається під час тестування алгоритму (від 50%). Використання алгоритму направлено на відповіді, представлені в короткій формі (до 10 слів) із можливими орфографічними помилками.

Висновки. Запропоновано обчислювальну технологію, яка передбачає, що зразок правильної відповіді, яка зберігається в базі даних та надана під час процедури опитування, розбиваються на окремі слова. Після цього проводиться пошук збігів за словами між еталоном та відповіддю, для чого застосовується модифікація методу динамічного програмування – алгоритм Хешберга. Далі відбувається знаходження загального показника подібності для всього речення шляхом додавання LCS усіх слів.

Подальші дослідження будуть спрямовані на тестування та удосконалення обчислювальної технології, застосування різних модифікацій алгоритмів динамічного програмування з метою підвищення ефективності обчислення LCS.

Список літератури:

1. Кузьма К.Т. Аналіз методів перевірки відповіді в системах тестування, поданої в текстовій формі. *Науковий журнал «Вчені записки ТНУ імені В.І. Вернадського. Серія: Технічні науки».* 2018. Том 29(68) № 1. Частина 1. С. 163–167.
2. Кузьма К.Т. Алгоритм перевірки відповіді в системах тестування, поданої у текстовій формі / К.Т. Кузьма. *Геометричне моделювання та інформаційні технології : науковий журнал*. № 2(6), жовтень 2018. Миколаїв: МНУ імені В.О. Сухомлинського, 2018. С. 30–33.
3. Лесько О.М., Рогушина Ю.В. Использование онтологий для анализа семантики естественно-языковых текстов. *Проблеми програмування.* 2009. № 3. С. 59–65.
4. Комарницька О.І., Ваколюк Т.В. Алгоритм нечіткого семантичного порівняння текстової інформації. *Збірник наукових праць Військового інституту Київського національного університету ім. Т. Шевченка*. Київ, 2013. № 39. С. 163–168.
5. Палагин А.В., Петренко Н.Г. К проектированию онтологоуправляемой информационной системы с обработкой естественно-языковых объектов. *Математичні машини і системи*. 2008. № 2. С. 14–23.
6. Катеринчук І.С., Рачок Р.В., Кравчук В.В., Кулик В.М. Інтелектуальна система автоматизованого контролю знань студентів вищих навчальних закладів. *Інформаційні технології в освіті :* збірник наукових праць. 2009. Вип. 4. Херсон : Вид-во ХДУ. С. 139–147.
7. Гула А.Ю., Игнатенко А.П., Чадюк А.В. Задачи идентификации физических и юридических лиц в хранилищах данных. Матеріали Шостої міжнародної науково-практичної конференції з програмування

УкрПРОГ 2008, 27-29 травня 2008 р. Київ. URL: http://dspace.nbuv.gov.ua/bitstream/handle/123456789/1464/%e2%84%962-3_2008_Ignatenko.pdf?sequence=1 (дата звернення: 30.12.2019).

8. Stephen Graham A. String Searching Algorithms. Lecture Notes Series On Computing. Vol. 3. London : World Scientific. 1994. 256 p.

9. Navarro G. A guided tour to approximate string matching. *ACM Computing Surveys*. 2001. 33(1): 31–88. P. 31–88. URL: https://www.dcc.uchile.cl/~gnavarro/ps/acmcs01.1.pdf (дата звернення: 30.12.2019).

10. Soundex. URL: http://en.wikipedia.org/wiki/Soundex (дата звернення: 30.12.2019).

11. Hirschberg D.S. A linear space algorithm for computing maximal common subsequences. *Communications of the ACM*. 1975. № 18(6). Pp. 341–343.

12. Смит Б. Методы и алгоритмы вычислений на строках: пер. с англ. Москва : ООО «И.Д. Вильямс», 2006. 496 с.

Kuzma K.T., Melnik O.V. COMPUTING TECHNOLOGY FOR CHECKING ANSWERS IN TESTING SYSTEMS

Actuality of the research is due to the rapid development of distance education, intellectualization of the process of testing answers in testing systems. According to the results of research of string comparison methods, the possibility of their application in testing systems to establish the level of correspondence of a short text answer with a standard (correct answer) has been determined.

Computing technology (CT) for verifying the short answer in testing systems submitted in the form of text was proposed. CT contains four basic stages: normalization of response and correct answer (make in lower case, all characters other than letters, numbers and space are deleted); words highlighting word selection (pronouns, adjectives are removed from sentences); comparing words of the response and correct answer by calculating the longest common subsequence, using a modification of the dynamic programming method, the Hirschberg's algorithm; finding a match for the entire sentence by adding individual word's LCS. The state of the similarity of the response with the reference is selected during the testing of the algorithm. This approach assumes that the greater the longest common subsequence, the greater the relevance of the correct answer.

By the use of dynamic programming methods, short answers that contain inaccuracies in the formulation are compared with the correct answer, which allowed us to solve the problem of verifying the answer presented in natural language.

Computational technology for validation of the answer, submitted in an arbitrary text form, has been developed for automation of the process of testing in systems of type "question – short open answer". It was determined that the algorithms of "approximate" string comparison which are related to the tasks of "fuzzy" string search are required further research.

Key words: *"fuzzy" search task, Hirschberg's algorithm, string comparison methods, testing systems, dynamic programming methods, longest common subsequence.*