УДК 004.021:81’32

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

АНАЛІЗ МЕТОДІВ ПЕРЕВІРКИ ВІДПОВІДІ В СИСТЕМАХ ТЕСТУВАННЯ, ПОДАНОЇ В ТЕКСТОВІЙ ФОРМІ

У роботі досліджено методи перевірки відповіді, поданої в довільній текстовій формі, проаналізовано їхні переваги та недоліки, що дозволило визначити найбільш ефективні й перспективні з погляду практичного застосування в системах тестування. Під час розгляду алгоритмів «точного» порівняння відповіді із правильним зразком (патерном) здійснено їх аналіз за базовими характеристиками. Відзначені обмеження застосування методів «точного» порівняння рядків у задачах перевірки відповідей, поданих у довільній текстовій формі. Визначено, що подальшого дослідження потребують саме алгоритми «приблизного» порівняння із множинними патернами шляхом побудови недетермінованого кінцевого автомата, який би містив усі патерни. Завдяки використанню регулярних виразів здійснюється порівняння з альтернативними, кратними підрядками, що дозволяє вирішувати задачу перевірки відповіді, поданої в текстовій формі, як задачу «нечіткого» пошуку.

Ключові слова: *відповідь, подана в текстовій формі, методи порівняння рядків, системи тестування, лінгвістичний аналіз тексту.*

Постановка проблеми. Наявні системи тестового контролю за своєю функціональністю суттєво обмежують можливості неформалізованої побудови тестових завдань. Сучасні програмні засоби, які використовуються для тестування, дозволяють будувати питання лише певних типів. Переважно це питання типу «одне питання – декілька варіантів відповідей, серед яких один правильний», «одне питання – декілька варіантів відповідей, серед яких декілька правильних», «зіставлення варіантів відповідей».

Недостатньо формалізоване подання відповіді на питання в довільній текстовій формі.

Тому для автоматизації перевірки відповіді, поданої в текстовому форматі природною мовою, необхідно розробити ефективну методику порівняння такої відповіді зі зразком (зразками) правильної відповіді.

Аналіз останніх досліджень і публікацій. У зв’язку з постійним зростанням потреб у застосуванні механізмів природної мови в атоматизованих інформаційних системах і людино-машинних системах особливого значення набули питання моделювання природної мови та мовлення. Це привело до розроблення різноманітних лінгвістичних моделей, що могли б розв’язати практичні завдання лінгвістики, а саме: інформаційний пошук, машинний переклад, розуміння природної мови тощо.

Досліджували, розробляли моделі та методи лінгвістичного аналізу тексту такі науковці: О. Комарницька, О. Лесько, О. Палагін, І. Катеринчук та інші [1–4].

Постановка завдання. Метою роботи є аналіз методів перевірки відповідей, поданих у довільній текстовій формі, для виявлення їхніх переваг і недоліків під час використання в системах тестування.

Виклад основного матеріалу дослідження. Вивчення й опис природних мов у контексті автоматизованих інформаційних систем потребує застосування математичних методів, серед яких: комбінаторні методи, методи математичної статистики, булевої алгебри, теорія графів, теорія нечітких множин; теорія ймовірності, методи штучного інтелекту (зокрема нейромережі).

Під час розгляду алгоритмів порівняння рядків тексту здійснено їх класифікацію на два основних види: «точного» порівняння зі зразком (патерном), «приблизного» порівняння з «патерном». Водночас шаблон (патерн) пошуку може бути одиночним або множинним.

Так, у роботі О. Комарницькою [1] розроблено метод нечіткого семантичного порівняння, який базується на алгоритмі «приблизного порівняння» із множинними патернами. Передбачається автоматизоване визначення лексичних одиниць тексту

з подальшим здійсненням морфологічного, синтаксичного, семантичного та прагматичного аналізу. Метод, розроблений у зазначеній роботі [1] для розпізнавання та виправлення слів, написаних із помилками (вставка, заміна, пропуск, транспозиція), базується на вдосконаленні метрики Левенштейна. Такий підхід передбачає, що, чим більшою є відстань між рядками, тим більшою є відмінність.

Оскільки в комп'ютері текстова інформація кодується числами, кожний текстовий рядок являє собою вектор у N-вимірному просторі, де N – кількість символів в рядку [4].

Функція *d (x, y)* для обчислення відстані між двома векторами x та y повинна мати такі властивості:

– невід'ємність: d (x, y) $\geq 0 \; \forall$ x, y;

– властивість нуля: d (x, y) = 0 $\Leftrightarrow$ x = y;

– симетричність: d (x, y) = d (y, x) $\forall$ x, y;

– нерівність трикутника: d (x, z) $\forall$ d (x, y) + d (y, z) $\forall$ x, y, z.

Відповідно до наведених властивостей є можливість побудувати багато різних метрик, однією з яких є евклідова метрика:

$$d(x, y) = \sqrt{\sum_i (x_i - y_i)^2} \; .$$

Проте для завдання оброблення текстової інформації така метрика не досить зручна. Кількість символів, які опитуваний використовує у відповіді на тестове питання, не є константою. Тому виникає потреба порівнювати рядки різної довжини, а отже, розмірності просторів, в яких вони перебувають.

Нехай S_1 та S_2 – два рядки, що мають довжину M та N відповідно над деяким алфавітом, тоді редакційну відстань (відстань Левенштейна) $d(S_1, S_2)$ можна розрахувати за такою рекурентною формулою:

$$d(S_1, S_2) = d(M, N), \text{ де}$$

$$D(i, j) = \begin{cases} 0; i = 0; j = 0 \\ i, j = 0, i > 0 \\ j, i = 0, j > 0 \\ \min \begin{pmatrix} D(i, j-1) + 1, \\ D(i-1, j) + 1, \\ D(i-1, j-1) + m(S_1[i], S_2[j]) \end{pmatrix}; j > 0; i > 0 \end{cases}$$

де $m(a, b)$ дорівнює нулю, якщо $a = b$, інакше одиниці. $\min(a, b, c)$ повертає найменший з аргументів.

Отже, зразок і відповідь розбиваються на окремі слова. Після чого проводиться пошук збігів за словами між зразком і відповіддю, для чого застосовується алгоритм Левенштейна. Перевагою методу є й те, що він дозволяє встановлювати обмеження на кількість можливих у відповіді помилок, що сприяє адекватному оцінюванню знань того, хто навчається.

Для формування загальної оцінки відповіді на тестові завдання використовується комплексний показник, у якому враховано наявність у відповіді слів, присутніх у зразку (зокрема й за умови нечіткості), відповідність структур зразка і відповіді (порядку слів).

У концептуальній моделі порівняння текстової інформації за змістом на етапах семантичного та прагматичного аналізу О. Комарницькою запропоновано також застосовувати моделі штучного інтелекту, зокрема нейромережі. Перший шар нейромережі містить дві групи нейронів. До цього шару вноситься вхідна інформація – відповідь і зразок. Основна обробка інформації реалізується в наступних шарах нейронів, в яких здійснюється семантичний аналіз відповіді. Для отримання результату перевірки використовується останній шар, який відображує ступінь ідентичності двох текстів за змістом. Перевагами використання нейромережі є універсальність. Незмінну за структурою нейромережу можна пристосувати для порівняння текстів у різних предметних сферах.

Базовими алгоритми «точного» порівняння рядків є:

1. Зіставлення рядків найпростішим порівняльним методом (алгоритм прямого пошуку).
2. Алгоритм Кнута-Морріса-Пратта.
3. Алгоритм Бойера-Мура.
4. Алгоритм Карпа-Рабіна.

Порівняння рядків полягає в тому, щоб знайти в тексті всі входження рядка-шаблону. Шаблон позначається x = x [0 .. m-1]; його довжина дорівнює m. Текст позначається за допомогою y = y [0 .. n-1]; його довжина дорівнює n. Обидва рядки побудовані за кінцевим набором символів, який називається алфавітом, з розміром, що дорівнює σ. Фундаментальні методи й алгоритми порівняння рядків містять роботи [5–7].

Характеристики алгоритмів «точного» порівняння рядків наведено в таблиці 1.

Методи «точного» порівняння рядків мають певні обмеження в застосуванні для рішення задачі перевірки відповіді, поданої в довільній текстовій формі, а саме:

1. Під час підготовки відповіді опитуваний може зробити помилку в словах, неправильно побудувати речення, вживати нестандартні скорочення й абревіатуру тощо.

2. Передбачають порівняння з одним патерном, тоді як необхідне порівняння із множиною патернів.

Для рішення задачі порівняння текстового рядка з набором патернів (множинний патерн) можна побудувати кінцевий автомат, який би включав всі патерни.

На основі недетермінованого кінцевого автомата NDFA (NDFA – non-deterministic finite automata) вирішується задача порівняння з регулярними виразами. Патерни, які є регулярними виразами, містять метасимволи | або *, які дозволяють проводити порівняння з альтернативними та кратними підрядками відповідно. Отже, патерн стає «приблизним». Застосування регулярних виразів під час перевірки відповіді, поданої природною мовою, дозволить реалізувати алгоритм нечіткого пошуку для урахування некоректно поданих слів у відповіді, що підвищить ефективність оцінювання.

«Недетермінованість» проявляється в тому, що з деяких станів (вершин) можливі кілька переходів (вихідних дуг), помічених символом нового рядка ε, і водночас визначення вихідної дуги неоднозначно. NDFA має єдиний початковий стан або джерело і тільки одну кінцеву вершину, яка відповідає поглинаючому стану або стоку.

Автомат $\mathrm{NDFA}(p)$, який відповідає регулярному виразу (p), представлено на рисунку 1.

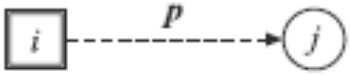

Рис. 1. Автомат $\mathrm{NDFA}(p)$

i та j – мітки початкового і поглинаючого станів відповідно, а пунктирна лінія позначає можливі шляхи від i до j.

Наведений нижче алгоритм NDFA використовує функцію trans для обчислення в даному рядку x усіх позицій i, для яких має місце збіг підрядків $x[i'..i]$ із непорожнім регулярним виразом p.

Алгоритм NDFA (знаходження всіх підрядків рядка x, що збігаються із заданим непорожнім регулярним виразом p; побудова $\mathrm{NDFA}(p)$ із поглинаючим станом w:

S ← trans(∅, ε)
for i ← 1 to n do
S ← trans(S, x [i])
if w ∈ S then
output i
S ← trans(S, ε)
if w ∈ S then
output i.

Функція trans (від *transition* – перехід) для заданої множини станів S та заданої літери $\lambda \in A$ (поточна вхідна літера) обчислює нову множину станів, які досягаються шляхом проходження дугами, позначеними літерою λ і виходять із вершин множини S.

Стан j автомата $\mathrm{NDFA}(p)$ називається λ-досяжним зі стану i, якщо виконуються такі умови:

– стан (вершина) i має вихідну дугу, яка позначається літерою λ;

– у $\mathrm{NDFA}(p)$ є орієнтований шлях від вершини i до вершини j, дуги якого позначені як $\lambda\varepsilon^k$ для деякого цілого $k \geq 0$;

– стан j не має вихідних дуг, які позначені як ε.

Дані умови цілком визначають функцію $\mathrm{trans}(\mathrm{S},\lambda)$. Водночас враховуються два особливих випадки:

– коли $S = \varnothing$ (порожня множина), функція $\mathrm{trans}(\mathrm{S},\lambda)$ інтерпретується як $\mathrm{trans}(\{0\},\lambda)$, де 0 – мітка початкового стану автомата NDFA (p);

Таблиця 1

Характеристики алгоритмів «точного» порівняння рядків

Алгоритм	Час на фазу попередньої обробки	Часова межа пошуку в середньому	Часова межа в найгіршому разі (пошук неперіодичного шаблону)	Витрати пам'яті	Базові переваги або недоліки
Прямого пошуку	–	$O(m{\cdot}n)$	$O(m{\cdot}(n{-}m{+}1))$	–	Мала ефективність
Кнута-Морріса-Пратта	$O(m)$	$O(m{+}n)$	$O(m{\cdot}(n{-}m{+}1))$	$\Theta(m)$	Універсальний алгоритм, якщо невідома довжина зразка
Бойера-Мура	$O(m{+}\sigma)$	$O(m{\cdot}n)$	$O((m{\cdot}(n{-}m{+}1){\cdot}\ m{+}\sigma)$	$\Theta(m{+}\ \sigma)$	Найбільш ефективний, швидкодія підвищується за збільшення зразка або алфавіту
Карпа-Рабіна	$O(m)$	$O(m{+}n)$	$O(m{\cdot}(n{-}m{+}1))$	$\Theta(m)$	Вимагає додаткової пам'яті, ефективний у разі $m{\geq}200$

– позначення $\mathrm{trans}(S,\varepsilon)$ інтерпретується як безліч усіх станів, які досягаються із множини S, якщо слідувати дугами, які виходять із цієї множини, позначеними символом нового рядка ε.

Для непорожнього регулярного виразу p, що містить m букв, та для заданого рядка $x = x[1..n]$ алгоритм, заснований на використанні NDFA, обчислює всі підрядки рядка x, що збігаються з патерном p, за час порядку $O(m \cdot n)$ із використанням пам'яті об'ємом $\Theta(m)$ [5, с. 346].

Для моделі NDFA, яка використовує бінарні вектори, доцільно використовувати два алгоритми для приблизного порівняння з патерном, один з яких заснований на алгоритмі BM (алгоритм Ву-Менбере), а другий запропонований Бейза-Ятсом і Наварро (BYN-Baeza-Yates, Navarro). Нова модифікація алгоритму BM використовується до функцій відстані будь-яких типів, а алгоритм BYN – тільки для просторових перетворень, Левенштейна і Хеммінга.

Алгоритми порівняння з регулярними виразами й із множинними патернами неможливо описати без використання поняття кінцевого автомата. Різні типи кінцевих автоматів вирішують свої завдання.

Висновки. Отже, враховуючи обмеження «точних» методів у задачах перевірки відповідей, представлених у довільній текстовій формі, подальшого дослідження потребують методи «нечіткого» порівняння, які базуються на використанні недетермінованого кінцевого автомата.

Список літератури:

1. Комарницька О., Ваколюк Т. Алгоритм нечіткого семантичного порівняння текстової інформації. Збірник наукових праць Військового інституту Київського національного університету ім. Тараса Шевченка. К., 2013. № 39. С. 163−168.

2. Лесько О., Рогушина Ю. Использование онтологий для анализа семантики естественно-языковых текстов. Проблеми програмування. 2009. № 3. С. 59−65.

3. Палагин А., Петренко Н. К проектированию онтологоуправляемой информационной системы с обработкой естественно-языковых объектов. Математичні машини і системи. 2008. № 2. С. 14−23.

4. Катеринчук І., Рачок Р., Кравчук В., Кулик В. 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.

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

6. 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: 30.01.2018).

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

АНАЛИЗ МЕТОДОВ ПРОВЕРКИ ОТВЕТА В СИСТЕМАХ ТЕСТИРОВАНИЯ, ПРЕДСТАВЛЕННОГО В ТЕКСТОВОЙ ФОРМЕ

В работе исследованы методы проверки ответа, представленного в произвольной текстовой форме, проанализированы их преимущества и недостатки, что позволило определить наиболее эффективные и перспективные с точки зрения практического применения в системах тестирования. При рассмотрении алгоритмов «точного» сравнения ответа с правильным образцом (паттерном) осуществлен их анализ по базовым характеристикам. Отмечены ограничения применения методов «точного» сравнения строк в задачах проверки ответов, представленных в произвольной текстовой форме. Определено, что дальнейшего исследования требуют именно алгоритмы «приблизительного» сравнение с множественными паттернами путем построения недетерминированного конечного автомата, который бы включал все паттерны. За счет использования регулярных выражений осуществляется сравнение с альтернативными, кратными подстроками, что позволяет решать задачу проверки ответа, представленного в текстовой форме, как задачу «нечеткого» поиска.

Ключевые слова: *ответ, представленный в текстовой форме, методы сравнения строк, системы тестирования, лингвистический анализ текста.*

ANALYSIS OF THE METHODS OF VERIFICATION THE ANSWER IN TESTING SYSTEMS, SUBMITTED IN A TEXT FORM

The methods of checking the answer submitted in an text form, their advantages and disadvantages, which allowed to determine the most effective and promising for practical use in testing systems have been considered. While investigating algorithms of "strict" comparison the answer with a correct pattern, their basic characteristics have been analyzed. Restrictions in use the "strict" comparison strings methods in tasks of verification the answers submitted in an text form have been noted. It was determined that further researching require the algorithms of "approximate" comparison with multiple patterns by constructing a nondeterministic finite automata that would include all the patterns. By using regular expressions, comparison with alternative, multiple substrings is performed, which allows to solve the task of verifying the answer submitted in text form as a task of "fuzzy" search.

Key words: *answer submitted in text form, methods of string comparison, testing system, linguistic analysis of text.*