УДК 621.39

О. М. ТКАЧЕНКО, О. Ф. ГРІЙО ТУКАЛО

Вінницький національний технічний університет, м. Вінниця

МЕТОД ПІДВИЩЕННЯ ШВИДКОСТІ ПОШУКУ ФРАГМЕНТУ АУДІОЗАПИСУ ІЗ ЗАСТОСУВАННЯМ KD-ДЕРЕВ

Анотація. Стаття присвячена розв'язанню задачі ідентифікації музичного твору за коротким фрагментом. В статті запропоновано метод пошуку найближчого шаблону 5с аудіофрагменту на основі kd-дерева, що дозволив значно зменшити складність обчислень порівняно з повним пошуком. Для виконання швидкого пошуку на основі kd-дерева в попередньо сформованому корпусі шаблонів розроблено алгоритмічне та програмне забезпечення. Запропоновано аналітичне співвідношення для оцінювання близькості невідомого фрагменту з шаблонами, яке базується на обчисленні відстані від фрагменту до заданої кількості найближчих центроїдів.

Ключові слова: швидкий пошук, kd-дерево, Евклідова відстань, ідентифікація за фрагментом аудіозапису, параметризація, мел-частотні кепстральні коефіцієнти, кластерний аналіз.

Аннотация. Статья посвящена решению задачи идентификации музыкального произведения по короткому фрагменту. В статье предложен метод поиска ближайшего шаблона 5с аудиофрагмента на основе kd-дерева, который позволил значительно уменьшить сложность вычислений по сравнению с полным поиском. Для выполнения быстрого поиска на основе kd-дерева в предварительно сформированном корпусе шаблонов разработано алгоритмическое и программное обеспечение. Предложено аналитическое соотношение для оценки близости неизвестного фрагмента с шаблонами, основанное на вычислении расстояния от фрагмента с заданным числом ближайших центроидов.

Ключевые быстрый поиск, kd-дерево, Евклидовое расстояние, идентификация по фрагменту аудиозаписи, параметризация, мел-частотные кепстральные коэффициенты, кластерный анализ.

Abstract. The article is devoted to solving the problem of musical identification by the short fragment. The method of the nearest template search for audio fragment (5s) based on kd-tree was proposed in the paper, which allowed to reduce the computational complexity significantly compared to full search. To perform a quick search based on kd-tree in pre-formed body of templates algorithms and software were developed. Analytical relation for evaluating the proximity between the unknown fragment and templates was proposed, which is based on calculating the distance from the fragment to the specified number of the nearest centroids.

Key words: quick search, kd-tree, Euclidean distance, audio fragment identification, parameterization, mel-frequency cepstral coefficients, cluster analysis.

Вступ

В сучасних комп’ютерних мережах основний об’єм трафіку припадає на мультимедійну, зокрема, аудіоінформацію. Зростання обсягу мультимедійної інформації, що передається і обробляється в комп’ютерних системах, зумовила необхідність автоматизації процесів аналізу і пошуку даних. Тому в сучасних системах обробки аудіоінформації виникає необхідність автоматичного швидкого пошуку музичних творів на основі аудіоконтенту у базах даних великого розміру, розміщених на віддалених серверах. Суть пошуку музичного твору на основі аудіоконтенту полягає в тому, щоб автоматично отримувати файли аудіозаписів музичних творів, подібних до заданого аудіозапису під час запиту. Враховуючи великі обсяги аудіоінформації в БД, велике значення має швидкість пошуку. З огляду на це в статті пропонується метод пошуку фрагменту аудіо в корпусі з застосуванням kd-дерев, що дозволяє значно збільшити швидкість пошуку.

1 Мета та задачі статті

Метою даної статті є зменшення складності обчислень під час пошуку фрагменту аудіозапису в сформованому корпусі шаблонів музичних творів.

Для зменшення складності обчислень в процесі автоматичного пошуку фрагмента музичного твору необхідно розв’язати такі задачі:

1. Обрати параметри, які дозволили б однозначно та компактно описати музичний твір.
2. Створити швидкий та надійний метод порівняння за обраними параметрами вхідного фрагменту музичного твору та попередньо створених еталонів (шаблонів) корпусу.
3. Мінімізувати тривалість фрагменту, що дозволяє ідентифікувати музичний твір.
4. Перевірити відповідність теоретичних припущень та експериментальних результатів.

Результатом розпізнавання буде шаблон БД з мінімальним розходженням відносно вхідного аудіозапису. Далі в роботі вважається, що аудіозапис, який треба ідентифікувати, точно збігається з одним із шаблонів корпусу.

2 Вибір математичної моделі аудіосигналу (MFCC)

Схема визначення відповідності музичних творів на основі контенту базується на використанні аудіофайлу для побудови моделі аудіосигналу. Порівнювати безпосередньо звукові сигнали в часовій області — не ефективно, тому відліки аудіо сигналу ділять на невеликі фрагменти (фрейми) тривалістю 10 – 30 мс, для яких характеристики сигналу залишаються відносно стійкими (випадковий процес є стаціонарним). Для кожного фрейму виконується спектральний аналіз, на основі якого (тим чи іншим чином) обчислюється значення вектора параметрів (параметризація). Багато різних параметрів запропоновано в літературі [1, 2]. В роботі як параметри обрані мел-частотні кепстральні коефіцієнти (MFCC – Mel Frequency Cepstral Coefficients), які вперше було запропоновано використовувати в системах розпізна-

вання мовлення та диктора [3], в подальшому вони отримали широке застосування в процесі інформаційного пошуку музики (MIR) [4, 5], зокрема, під час класифікації за жанрами, визначенні подібності аудіо тощо. Основні етапи отримання векторів параметрів MFCC показано на рис. 1.

Рисунок 1 – Етапи отримання параметрів MFCC

Обравши MFCC як параметри, ми отримуємо опис музичного твору у вигляді файлу з параметрами MFCC, що проілюстровано на рис. 2.

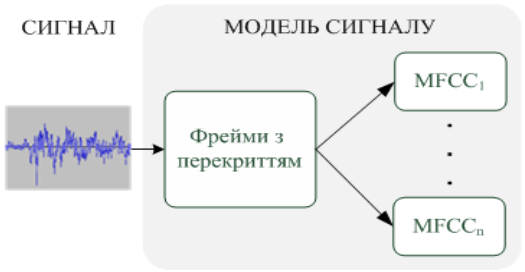

Рисунок 2 – Математична модель аудіосигналу - MFCC

Таким чином, якщо тривалість музичного твору в середньому складає 3хв (180с), частота дискретизації аудіофайлу – 44,1кГц, довжина фрейму – 20мс з перекриттям 0,5 фрейма, то такий середньостатистичний музичний твір (рис. 3) можна описати приблизно 8 млн. відліків, або, з врахуванням перекриття фреймів, приблизно 18тис. фреймів (9000 фреймів*2–1), кожен з яких описується вектором параметрів MFCC розмірності 13.

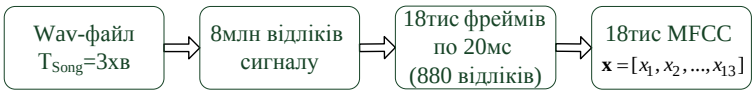

Рисунок 3 – Опис wav-файлу (3хв)

Тобто параметризація дозволяє зменшити кількість інформації, необхідної для опису музичного твору, в десятки разів:

$$\frac{7\ 938\,000\ відліків}{17999\ MFCC \cdot 13} = \frac{7\,938\,000}{233\,987} \approx 34\ рази$$

Таким чином, коефіцієнти MFCC є компактним представленням спектральної обвідної, що під час розпізнавання музичного твору дозволяє успішно замінити мільйони відліків аудіофайлу.

3 Порівняння невідомого музичного твору з шаблонами БД за приведеною відстанню (ПВ)

Після вибору параметрів для опису музичних творів, необхідно перейти до порівняння невідомого музичного твору з шаблонами корпусу та визначення шаблону (власного музичного твору), розходження з яким буде мінімальним. Для того, щоб ідентифікувати невідомий аудіозапис, необхідно мати критерій порівняння. Як правило, таким критерієм є відстань D. Підхід порівняння за обраними параметрами має забезпечити високий рівень розрізнення власного шаблону $\tilde{\mathbf{X}}$ та шаблонів інших творів $\tilde{\mathbf{Y}}$. Тобто в ре-

зультаті порівняння невідомого музичного твору $\mathbf{X}$ з шаблоном власного твору похибка між ними має бути мінімальною, і якомога більшою – для шаблонів решти творів:

$$\begin{cases} D(\mathbf{X},\tilde{\mathbf{X}}) \to \min \\ D(\mathbf{X},\tilde{\mathbf{Y}}) \to \max \end{cases} \Rightarrow D(\mathbf{X},\tilde{\mathbf{X}}) << D(\mathbf{X},\tilde{\mathbf{Y}}), \tag{1}$$

де $\tilde{\mathbf{X}}$ - множина векторів параметрів шаблону власного музичного твору; $\tilde{\mathbf{Y}}$ - множина векторів параметрів шаблону іншого музичного твору.

Загальну схему ідентифікації музичного твору наведено на рис. 4.

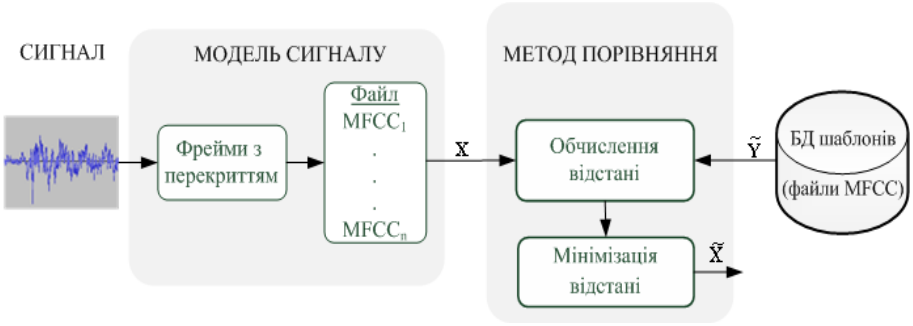

Рисунок 4 – Загальна схема ідентифікації музичного твору

Найпростішим і очевидним підходом для визначення близькості між наборами параметрів MFCC невідомого музичного твору та еталонів БД є порівняння MFCC на основі найбільш розповсюдженої Евклідової метрики, точніше квадрату Евклідової відстані $D_{Eu}{}^2$ (щоб надати велику вагу більш віддаленим об’єктам). Формула незваженої Евклідової відстані між вектором параметрів музичного твору, який треба ідентифікувати, $\mathbf{x} = (x_1, x_2, ..., x_d)$ та вектором шаблону БД $\tilde{\mathbf{y}} = (\tilde{y}_1, \tilde{y}_2, ..., \tilde{y}_d)$:

$$D_{Eu}{}^2(\mathbf{x}, \tilde{\mathbf{y}}) = \sum_{i=1}^{d} (x_i - \tilde{y}_i)^2. \tag{2}$$

Відповідно відстань (похибку) між файлами параметрів невідомого твору та шаблону БД можна знайти як суму відстаней:

$$D(\mathbf{X},\tilde{\mathbf{Y}}) = \sum D_{Eu}{}^2(\mathbf{x}_j,\tilde{\mathbf{y}}_j). \tag{3}$$

В ідеальному випадку при такому підході відстань між файлами параметрів MFCC одного і того ж музичного твору буде рівна нулю, для різних творів – відмінною від нуля:

$$\begin{cases} D(\mathbf{X},\tilde{\mathbf{X}}) = 0, \\ D(\mathbf{X},\tilde{\mathbf{Y}}) > 0. \end{cases} \tag{4}$$

Проте варто зауважити, що навіть для одного і того ж музичного твору аудіозаписи можуть відрізнятися, наприклад: на початку запису може йти тиша, мелодія іншого музичного твору тощо; записи можуть мати різний темп, тривалість. Це означає, що у разі зсуву фреймів в часі відстань до власного шаблону $D(\mathbf{X},\tilde{\mathbf{X}}) \neq 0$, тобто умова розрізнення власного музичного твору (4) не виконується, отже, безпосереднє порівняння файлів параметрів за Евклідовою відстанню не підходить для задачі ідентифікації власного шаблону.

В цьому випадку придатним для ідентифікації музичного твору є алгоритм динамічної трансформації шкали часу (DTW - Dynamic Time Warping), який дозволяє працювати з послідовностями векторів, що мають певний зсув в часі [6]. Однак використання DTW призведе до зростання кількості операцій порівняння N_{op}, що є неприйнятним.

Очевидно, що в більшості випадків музичний твір характеризується певною періодичністю, що полягає в наявності ідентичних або дуже схожих за текстом та характером мелодії фрагментів. Відповідно можна говорити про надлишковість даних, якими описується музичний твір, і можливість скоротити кількість параметрів для його опису. З огляду на це доцільним є застосування методів кластерного аналі-

зу для формування корпусу шаблонів музичних творів, що буде описано в розділі 4. Використання кластеризації для формування еталонів корпусу, дозволить зменшити обсяги пам'яті, необхідні для їх зберігання.

Основні етапи порівняння файлів параметрів невідомого музичного твору з певним шаблоном БД:

1. Пошук мінімальної евклідової відстані $D\min^2$ між поточним вектором параметрів $\mathbf{x} = (x_1, x_2, ..., x_d)$ з множини параметрів $\mathbf{X} = \{\mathbf{x}\}, |\mathbf{X}| = n$ музичного твору, який треба ідентифікувати, та множиною векторів-центроїдів $\tilde{\mathbf{Y}} = \{\tilde{\mathbf{y}}\}, |\tilde{\mathbf{Y}}| = m$, $\tilde{\mathbf{y}} = (\tilde{y}_1, \tilde{y}_2, ..., \tilde{y}_d)$ шаблону:

$$D\min^2 = \underbrace{\min}_{j}(D_{Eu}{}^2(\mathbf{x}, \tilde{\mathbf{y}}_j)) = \underbrace{\min}_{j}(\sum_{i=1}^{d}(x_i - \tilde{y}_{ji})^2), j = \overline{1,m}. \quad (5)$$

2. Обчислення оцінки відстані в цілому до шаблону як суми квадратів мінімальних відстаней $D\min^2$:

$$D(\mathbf{X}, \tilde{\mathbf{Y}}) = \sum_{l=1}^{n} D\min^2 = \sum_{l=1}^{n} \underbrace{\min}_{j}(\sum_{i=1}^{d}(x_i - \tilde{y}_{ji})^2), j = \overline{1,m}. \quad (6)$$

Разом з тим, оскільки музичні твори мають різну тривалість, тобто кожний твір характеризується власною кількістю фреймів, представлених параметрами MFCC. Під час кластеризації обчислюється однакова кількість центроїдів для усіх шаблонів. Це призводить до того, що різні твори знаходяться в нерівних умовах, тобто для творів, тривалість яких більша, початкова похибка (між файлами того ж твору до і після кластеризації) теж буде більшою, оскільки в цьому випадку на кожен кластер буде припадати більше векторів параметрів. Позбутися цього можна за рахунок ділення відстані до шаблону, визначеної в формулі (6), на кількість фреймів музичного твору n. Назвемо цю величину приведеною відстанню (ПВ - $D_{ПВ}$) музичного твору:

$$D_{ПВ} = \frac{D(\mathbf{X}, \tilde{\mathbf{Y}})}{n} = \frac{\sum_{l=1}^{n} D\min^2}{n} = \frac{\sum_{l=1}^{n} \min(\sum_{i=1}^{d}(x_i - \tilde{y}_{ji})^2)}{n}, j = \overline{1,m}, \quad (7)$$

Як можна побачити з формули (7) характеристика $D_{ПВ}$ не залежить від кількості фреймів (тривалості запису). Таким чином, її можна використовувати як критерій прийняття рішення як для запису в цілому, так і для його окремого фрагменту, що було теоретично обгрунтовано в одній з попередніх статей [7] та визначено тривалість фрагменту музичного твору в 5с (500 фреймів, або 500 векторів параметрів MFCC) мінімально достатньою для ідентифікації аудіозапису.

4 Створення корпусу шаблонів музичних творів застосовуючи кластерний аналіз

Задача кластеризації даних є важливим елементом загальної проблеми обробки даних. Кластеризацію часто використовують, зокрема, під час статистичного аналізу даних, векторної квантизації, розпізнавання образів тощо. Кластеризація — це поділ множини вхідних даних (в нашому випадку векторів параметрів MFCC) на групи (кластери) за мірою «схожості» один на одного, тобто таким чином, щоб кожен кластер містив найбільш схожі об'єкти, а об'єкти різних кластерів відрізнялися між собою. Задачу кластеризації можна сформулювати так: заданий набір з *n* векторів, кожен з яких має розмірність *d*, необхідно розбити на підмножини відповідно до заданого критерію оптимізації. Як правило, таким критерієм є мінімізація спотворення $e_i^2 \to \min$. Існують різні шляхи оцінювання спотворення, але в більшості прикладних реалізацій використовують суму середньоквадратичних Евклідових відстаней між центром кластеру (центроїдом) $\mathbf{C}_i$ і векторами параметрів, які до нього належать $\mathbf{X}_i = \{\mathbf{x}\}, \mathbf{X}_i \subset \mathbf{X}$ [8, 9], тобто:

$$e_i^2 = \{\mathbf{c}_i : \sum_{j=1}^{N_i} D_{Eu}{}^2(\mathbf{x}_j, \mathbf{c}_i) \mid \mathbf{x} \in \mathbf{X}_i \le \sum_{j=1}^{N_i} D_{Eu}{}^2(\mathbf{x}_j, \mathbf{c}) \mid \mathbf{x} \in \mathbf{X}_i\},$$
$$\mathbf{X}_i \subset \mathbf{X}, \forall \mathbf{c} \in \mathbf{X} \setminus \mathbf{X}_i, e_i^2 \to \min.$$

де N_i – кількість точок, що належать центроїду $\mathbf{C}_i$.

Таким чином, шаблони музичних творів в БД можна описати центроїдами кластерів параметрів MFCC, показаних на рис. 5 для двомірного випадку у вигляді кіл.

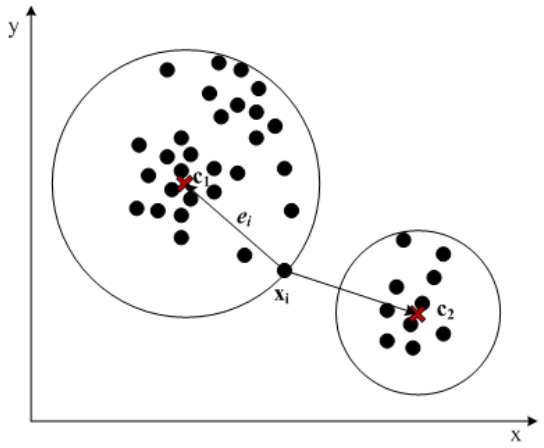

Рисунок 5 – Розбиття векторів параметрів на кластери

З рис. 5 видно, що між кожним центроїдом $\mathbf{c}_i$ та векторами, які належать до нього, є похибка e_i^2, що є сумою квадратів відстаней між ними. Звідси випливає, що відстань між файлами параметрів, що описують один і той же музичний твір до кластеризації і після (навіть якщо аудіозаписи були ідентичними), буде додатною і рівною величині сумарної похибки кластеризації E^2.

Отже, сформований корпус шаблонів музичних творів для ідентифікації невідомого фрагменту аудіозапису, фактично являє собою єдиний текстовий файл, що містить центроїди MFCC усіх шаблонів музичних творів.

5 Застосування kd-дерев для пошуку фрагменту музичного твору в корпусі шаблонів

Найпростіше вирішення загальновідомої задачі пошуку найближчого сусіда, а в нашому випадку - найближчого центроїда в корпусі музичних творів, - це обчислити відстань між вектором невідомого фрагменту музичного твору, який потрібно ідентифікувати, та усіма векторами-центроїдами шаблонів. Час виконання повного пошуку пропорційний $O(dN)$, де N - кількість центроїдів-векторів в корпусі (елементів дерева), d - розмірність векторів. Однак застосування такого підходу для пошуку у великих базах даних стає неможливим. Тому для зменшення складності обчислень під час пошуку фрагменту аудіозапису в сформованому корпусі було застосовано, мабуть найпростіший, спосіб розбиття простору - k-вимірне дерево (kd-дерево).

Тобто центроїди шаблонів корпусу було упорядковано на основі kd-дерева. kd-дерево - це структура даних для впорядкування точок в k-вимірному просторі. kd-дерево - особливий вид незбалансованих бінарних дерев пошуку, в якому кожна вершина задає розбиття простору на два підпростори деякою площиною, що проходить через неї вздовж осей (розмірностей) даних [10]. У kd-дереві крім кореневої присутні два типи вершин: термінальні (листи) та нетермінальні (вузли). Впорядкування векторів-центроїдів корпусу на основі kd-дерева виконується дуже швидко, оскільки розбиття простору відбувається лише вздовж осей даних, відповідно при цьому не потрібно обчислювати d-вимірні відстані.

Отже, використання kd-дерев дозволяє суттєво зменшити кількість вимірювань, необхідних для пошуку найближчого центроїда в корпусі. Однак пошук по kd-дереву не гарантує знаходження дійсно найближчого вектора-центроїда згідно з формулою (1), оскільки математично задача пошуку найближчого елемента в структурі kd-дерева формулюється так: дано деякий багатовимірний вектор $\mathbf{x}$; необхідно знайти вершину kd-дерева v', щоб виконувалась умова:

$$d(\mathbf{v}', \mathbf{x}) = \min\{d(\mathbf{v}_i, \mathbf{x})\}, i = (\overline{1, n}), \ \mathbf{v}_i \in \mathbf{V}, \tag{8}$$

де $\mathbf{x}$ – вектор, для якого виконується пошук найближчого центроїда-вектора в корпусі; $\mathbf{v}'$ – вершина дерева, відстань до якої найменша [10].

В роботі [11] запропоновано удосконалену процедуру пошуку по kd-дереву, яку застосовано для пошуку найближчого центроїда у корпусі шаблонів в процесі ідентифікації фрагменту музичного твору. Ілюстрацію пошуку найближчого центроїда у корпусі, впорядкованому на основі kd-дерева, для двовимірного випадку показано на рис. 6.

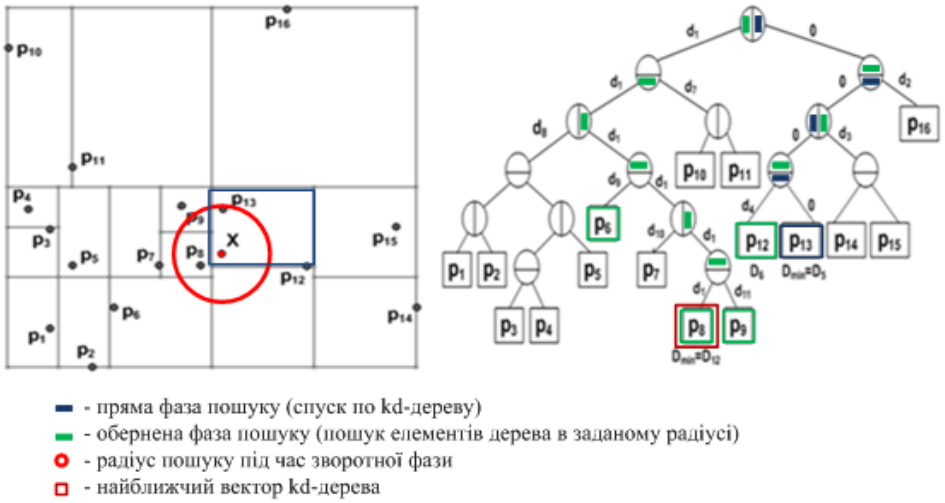

Рисунок 6 – Ілюстрація пошуку найближчого центроїда по kd-дереву для двовимірного випадку

Щоб забезпечити знаходження найближчого вектора, пошук, крім прямої фази пошуку (спуску по дереву), повинен мати також обернену (пошук елементів дерева в заданому діапазоні (радіусі)). Під час прямого пошуку фіксуються всі відстані до вузлів d_i. Пряма фаза завершується обчисленням відстані $D_{\min} = D_k$ до відповідної термінальної вершини (на рис. 6 - прямокутника), в якій лежить невідомий вектор, а також потенційно найближчий елемент дерева (центроїд в корпусі), що задає радіус пошуку під час оберненої фази. Після цього починається обернена фаза пошуку, при цьому обчислюються відстані d_i лише до тих вузлів дерева, які можуть забезпечити виконання $d_i < D_{\min}$. Якщо для відповідного листа виконується умова $D_k < D_{\min}$, радіус пошуку коригують $D_{\min} = D_k$ [11, 12].

Розглянута процедура пошуку гарантує знаходження найближчого вектора згідно з формулою (1), проте потребує більшої кількості вимірювань відстані, ніж $\log_2 n$. З метою скорочення часу в [12] запропоновано для обчислення відстані до вузлів скористатися більш ефективною формулою

$$D_k^{\,2} = (D_k^{\,2})' - (p_k^{'} - x_k)^2 + (p_k - x_k)^2, \tag{9}$$

де $(D_k^{\,2})'$ - раніше обчислена відстань до певного вузла дерева.

Застосування (9) можливе завдяки тому, що на кожному кроці просування по дереву змінюється лише одна з координат. Обчислення відстані згідно (9) потребує лише одного множення та трьох додавань, оскільки $(p_k^{'} - x_k)^2$ можна зберігати у структурі даних дерева. Обчислення відстані до листів доводиться виконувати за формулою (2).

Відзначимо, що пошук на основі kd-дерева є дуже ефективним лише для невеликих розмірностей ($d < 20$) Для $d < 20$ середній час пошуку найближчого вектора в корпусі, впорядкованого на основі kd-дерева, в середньому зростає пропорційно $O(h) = O(d \cdot \log(N+1))$, де h - висота дерева [11]. Для більших розмірностей обчислювальна складність може досягати $O(d \cdot N)$.

Наведена процедура пошуку забезпечує отримання не одного найближчого вектора, а деякої множини, упорядкованих за зростанням відстані згідно формули (2). Це можливо за рахунок того, що дані про відстань до вже пройдених термінальних вершин зберігаються, і одночасно здійснюється їх упорядкування за зростанням відстані до невідомого вектора. Завдяки цьому додаткове знаходження декількох найближчих векторів не вимагає багато часу.

6 Схема прийняття рішення за приведеною відстанню

Розглянемо стратегію пошуку найближчого шаблону в корпусі для невідомого фрагменту (ідентифікації фрагменту): Для кожного фрейму (вектора) $\mathbf{x}_l$ невідомого фрагменту:

1. Виконується швидкий пошук в упорядкованому на основі kd-дерева корпусі шаблонів музичних творів, в процесі якого за ЕМ згідно з формулою (2) відбирається множина $\mathbf{C}$ векторів-центроїдів шаблонів (кандидатів на найближчий вектор), упорядкованих за зростанням відстані:

$$\mathbf{C} \subset \mathbf{T}, \mathbf{C} = \{\tilde{\mathbf{Y}}_1, \tilde{\mathbf{Y}}_2, \ldots, \tilde{\mathbf{Y}}_k\}, \left|\tilde{\mathbf{Y}}\right| = m, |\mathbf{C}| = k, |\mathbf{T}| = N = m \cdot Q, k \le N,$$

де k - кількість векторів шаблонів в списку найближчих до фрейму невідомого фрагменту, m - кількість центроїдів (векторів) шаблону, Q - загальна кількість шаблонів музичних творів в корпусі, N - загальна кількість центроїдів-векторів корпусу музичних творів.

Для кожного з k найближчих центроїдів множини $\mathbf{C}$ зберігаються такі дані: відстань до фрейму невідомого фрагменту $D_p{}^2(\mathbf{x}, \tilde{\mathbf{y}}_{qj})$ та індекс відповідного шаблону музичного твору i_{pq} в корпусі:

$$\mathbf{I} = \{i_{pq}\}, \mathbf{D} = \{D_p{}^2(\mathbf{x}_l, \tilde{\mathbf{y}}_{qj})\}, \quad D_p{}^2(\mathbf{x}_l, \tilde{\mathbf{y}}_{qj}) = \sum_{i=1}^{d} (x_i - \tilde{y}_i)^2, \; p = \overline{1,k}, l = \overline{1,n}, \; j = \overline{1,m}, q = \overline{1,Q}.$$

2. Відстань до кожного шаблону визначається згідно з правилом:

$$Distance_q = \begin{cases} \min(D_p{}^2), \forall q \in \mathbf{I} \\ D_p{}^2, \; p = k, \forall q \notin \mathbf{I} \end{cases} \tag{10}$$

В процесі виконання описаного вище алгоритму обчислюється сума відстаней по усім n фреймам невідомого фрагменту для кожного шаблону $\tilde{\mathbf{Y}}_q$.

$$D(\mathbf{X}, \tilde{\mathbf{Y}}_q) = \sum_{l=1}^{n} Distance_q. \tag{11}$$

На останньому кроці знаходиться мінімум приведеної відстані, визначеної раніше в формулі (7):

$$D(\mathbf{X}, \tilde{\mathbf{X}}) = \underbrace{\min}_{q} D_{ПВq} = \underbrace{\min}_{q} \frac{D(\mathbf{X}, \tilde{\mathbf{Y}}_q)}{n}. \tag{12}$$

Відповідно шаблон власного музичного твору $\tilde{\mathbf{X}}$ в корпусі як найближчий до невідомого фрагменту вважається визначеним.

7 Перевірка відповідності теоретичних припущень та експериментальних результатів

Для проведення експериментальних досліджень сформовано корпус з 1000 шаблонів музичних творів. Всі музичні твори мали формат wav (mono) з частотою дискретизації 44,1кГц. Попередньо з аудіофайлів було видалено тишу з початку та кінця записів. В процесі формування БД аудіозаписи шаблонів було поділено на фрейми по 20мс з перекриттям 10мс та для кожного фрейму обчислено вектор параметрів MFCC розмірності 13. Послідовності векторів параметрів MFCC, що описують музичні твори, було кластеризовано, використовуючи вдосконалений метод кластеризації k-середніх, запропонований у одній з попередніх робіт [13]. Після кластеризації кожен шаблон БД було представлено 1000 центроїдів MFCC (в середньому на кластер припадає близько 20 векторів). Фрагменти (тривалістю 5с) для ідентифікації обиралися випадковим чином з еталонів БД. Для виконання швидкого пошуку основі kd-дерева для визначення найближчого шаблону невідомого фрагменту розроблено алгоритмічне та програмне забезпечення.

Застосування для пошуку kd-дерева та оцінювання відстаней до шаблонів на основі виразу (10), що було описано в попередньому розділі, порівняно з повним пошуком, призводить до деякого зменшення міри розрізнення на основі відстаней шаблону власного музичного твору та решти шаблонів БД (показано на рис. 7), проте різниця залишається достатньою для прийняття правильного рішення.

Для підвищення надійності результатів пошуку за рахунок збільшення міри розрізнення між власним шаблоном та іншими можна збільшити кількість центроїдів шаблонів, тривалість фрагменту для розпізнавання або ж кількість найближчих центроїдів, отримуваних в результаті пошуку по kd-дереву. Останній варіант є найбільш прийнятним, оскільки не призводить до значного зростання часу пошуку. Розглядались результати для 20 і 50 найближчих кандидатів. В табл. 1 наведено для п'яти фрагментів відстані до власного шаблону (перший рядок, виділений кольором) та ще чотирьох найближчих шаблонів.

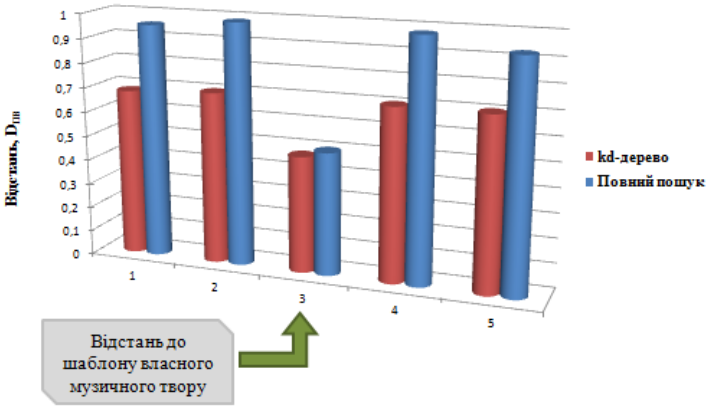

Рисунок 7 – Порівняння результатів повного пошуку і пошуку на основі kd-дерева

Таблиця 1 – Відстані до власного шаблону та чотирьох найближчих для *k*=20 і *k*=50

k	20					50				
№ фрагм.	1	2	3	4	5	1	2	3	4	5
$D_{ПВ}$	0,412236	0,454022	0,304221	0,363928	0,454186	0,425145	0,490213	0,324479	0,386068	0,474792
	0,753073	0,564241	0,434218	0,457813	0,615269	0,847361	0,63867	0,498193	0,520901	0,682864
	0,773479	0,566145	0,442772	0,459644	0,625111	0,863257	0,642791	0,508418	0,522412	0,70061
	0,779169	0,566525	0,4428	0,461187	0,625922	0,879997	0,644621	0,509206	0,523212	0,703468
	0,780288	0,566799	0,445319	0,461311	0,626101	0,885091	0,64546	0,51127	0,523405	0,703853

Отримані дані показують, що зі збільшенням кількості найближчих до 50, різниця між власним шаблоном та чужими зросла.

Нижче в табл. 2 та на рис. 8 показано як залежить час *t* пошуку найближчого шаблону за допомогою kd-дерева від кількості музичних творів корпусу *Q* для різної кількості найближчих *k*.

Таблиця 2. Залежність *t(Q,k)*

К-сть шаблонів	Час t, с	
	k=20	k=50
10	0,20	0,20
30	0,43	0,57
100	0,93	1,17
300	1,73	2,33
1000	3,00	4,10

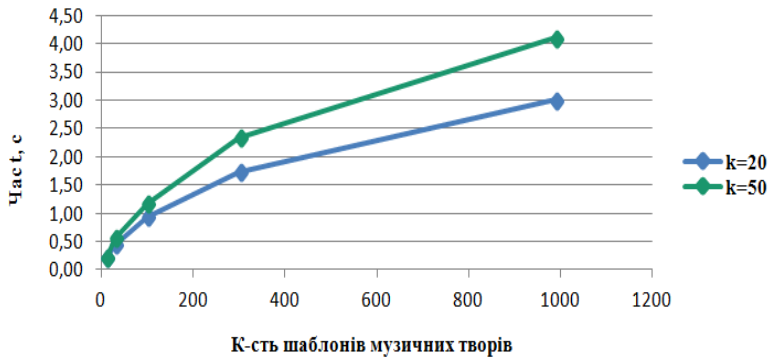

Рисунок 8 – Залежність *t(Q,k)*

На графіках видно, що час зростає повільніше, ніж кількість шаблонів в корпусі. Час пошуку найближчого шаблону (Windows7, Intel Core i5-4200M CPU @ 2.5GHz, 6GB RAM) в корпусі з 1000 шаблонів для 20 кандидатів становить 3с, що менше ніж тривалість фрагменту (5с) та значно менше порівняно з повним пошуком.

Щоб спрогнозувати як зросте складність обчислень при подальшому зростанні кількості шаблонів у корпусі, важливо перевірити наскільки експериментальні результати відповідають теоретичним оцінкам кількості операцій під час пошуку на основі kd-дерева. Тому нижче в табл. 3 наведено середню кількість операцій, що виконується в процесі пошуку найближчого шаблону на основі kd-дерева, поділену на розмірність простору $d = 13$.

Таблиця 3 – Залежність к-сті операцій під час пошуку від кількості шаблонів корпусу для *k*=20 і *k*=50

К-сть шаблонів	Середня к-сть операцій	
	k=20	k=50
10	83294,59	113123,43
30	154240,67	210651,01
100	292644,55	401930,94
300	515980,34	713548,06
1000	838046,80	1185758,00

Відповідно до результатів, наведених в табл. 3, при зростанні кількості шаблонів корпусу в 100 разів, кількість операцій зросла лише в 10 разів.

Теоретично залежність кількості операцій пошуку найближчого шаблону від кількості музичних творів корпусу повинна мати логарифмічний характер, тому для експериментальних кривих в табличному процесорі Excel було побудовано логарифмічні лінії тренду (рис. 9). Отриманий показник достовірності апроксимації R^2 як один з способів оцінювання правильності підібраного для прогнозування рівняння для 20 та 50 найближчих має значення $R^2 > 0{,}9$.

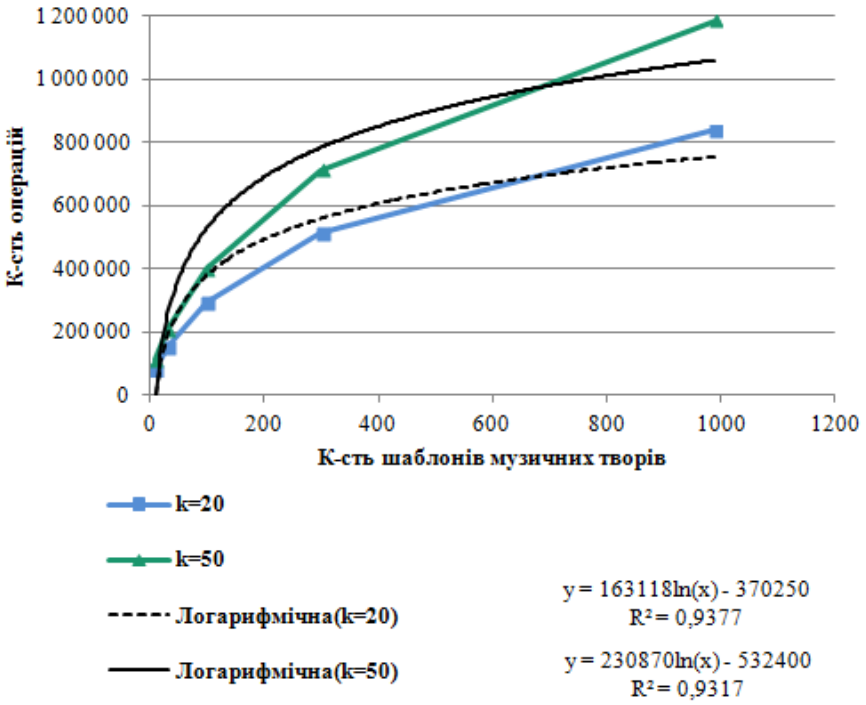

Рисунок 9 – Залежність кількості операцій пошуку найближчого шаблону від кількості музичних творів корпусу

Лінію тренду можна вважати визначеною точно, якщо $R^2 = 1$, тобто експериментальні результати не зовсім відповідають теоретичним оцінкам складності обчислень. Це може бути пов'язано з незбалансованістю kd-дерева, наявністю під час пошуку оберненої фази та пошук декількох найближчих.

Висновки

1. В статті запропоновано метод пошуку найближчого шаблону аудіофрагменту тривалістю 5с на основі kd-дерева, що дозволив значно зменшити складність обчислень порівняно з повним пошуком. Для виконання швидкого пошуку на основі kd-дерева в попередньо сформованому корпусі шаблонів розроблено алгоритмічне та програмне забезпечення.
2. Запропоновано аналітичне співвідношення для оцінювання близькості невідомого фрагменту з шаблонами, яке базується на обчисленні відстані від фрагменту до заданої кількості найближчих центроїдів.
3. За рахунок використання методу швидкого пошуку на основі kd-дерева було досягнуто зменшення часу на прийняття рішення та показано, що зі збільшенням кількості шаблонів в 100 разів середня кількість операцій зростає лише в 10 разів.

Список літератури

1. Wang Y. Multimedia content analysis using both audio and visual cues / Y. Wang, Z. Liu, and J. C. Huang // IEEE signal processing magazine. – 2000. – no. 17. – pp. 12–36.
2. Grosche P. Audio content-based music retrieval / P. Grosche, M. Müller, J. Serrà // Dagstuhl Follow-Ups Multimodal Music Processing. – V. 3. – Dagstuhl, Germany. – 2012. – pp. 157–175.
3. Ganchev T. Comparative evaluation of various mfcc implementations on the speaker verification task / T. Ganchev, N. Fakotakis, and G. Kokkinakis // Proceedings of 9th International Conference on Speech and Computer, SPECOM'05. – 2005. – pp. 191–194.
4. Logan B. A music similarity function based on signal analysis / B. Logan and A. Salomon // Proc. IEEE Int. Conf. Multimedia Expo. – 2001. – pp. 745–748.
5. Tzanetakis G. Musical genre classification of audio signals / G. Tzanetakis and P. Cook // IEEE Trans. Speech Audio Process. – No. 5. – V. 10. – 2002. – pp. 293–301.
6. Senin P. Dynamic time warping algorithm review / P. Senin – Honolulu, USA. – 2008.
7. Ткаченко О.М Підхід до оцінювання тривалості фрагмента для пошуку музичного твору за заданим шаблоном / О. М. Ткаченко, О. Ф. Грійо Тукало // Міжнародний науково-технічний журнал «Інформаційні технології та комп'ютерна інженерія». – №1. – Вінниця: ВНТУ. – 2014.
8. Gersho A. Vector Quantization and Signal Compression. / A. Gersho, R. M. Gray. – Boston: Kluwer Academic. – 1992. – 760 p.
9. Jain A. K. Algorithms for Clustering Data / A. K. Jain, R. C. Dubes. – Englewood Cliffs, N.J.: Prentice Hall. – 1988. – 334 p.
10. Moore A. An introductory tutorial on KD trees / Andrew Moore. – Pittsburgh, PA: Robotics Institute, Carnegie Mellon University. – 1991.
11. Friedman J. H. An algorithm for finding best matches in logarithmic expected time. / J. H. Friedman, J. L. Bentley, and R. A. Finkel // ACM Transactions on Mathematical Software, 3(3). – 1977. – pp. 209–226.
12. Arya S. Algorithms for fast vector quantization. / S. Arya and D. M. Mount. // Proc. of DCC '93: Data Compression Conference. – 1993. – pp. 381–390.
13. Ткаченко О.М. Метод кластеризації на основі послідовного запуску k-середніх з удосконаленим вибором кандидата на нову позицію вставки / О. М. Ткаченко, О. Ф. Грійо Тукало, О. В. Дзісь, С. М. Лаховець // Електронний журнал «Наукові праці ВНТУ». – №2. – В.2. – Вінниця: ВНТУ. – 2012.

Стаття надійшла: 25.11.2014.

Відомості про авторів

Ткаченко Олександр Миколайович – к.т.н., доцент кафедри обчислювальної техніки, Вінницький національний технічний університет, Хмельницьке шосе, 95, м. Вінниця, alextk1960@gmail.com.
Грійо Тукало Оксана Франсисківна – аспірант кафедри обчислювальної техніки, Вінницький національний технічний університет, Хмельницьке шосе, 95, м. Вінниця, xxmargoxx@gmail.com.