

# СИСТЕМЫ QUERY-BY-HUMMING: ОБЗОР ПОДХОДОВ И СХЕМА ПЛАТФОРМЫ ДЛЯ ЭКСПЕРИМЕНТОВ

И. А. БАЛТИЙСКИЙ<sup>1</sup>, С. И. НИКОЛЕНКО<sup>2</sup>

<sup>1</sup>Санкт-Петербургский государственный университет Информационных Технологий, Механики и Оптики, <sup>2</sup>Учреждение Российской академии наук Санкт-Петербургское отделение Математического института им. В.А. Стеклова РАН

<sup>1</sup>СПбГУ ИТМО, Кронверкский пр., 49, Санкт-Петербург, 197101,

<sup>2</sup>ПОМИ РАН, наб. р. Фонтанка, 27, Санкт-Петербург, 191023

<sup>1</sup><iosank@gmail.com>, <sup>2</sup><sergey@logic.pdmi.ras.ru>

---

УДК 004.89

*Балтийский И. А., Николенко С. И. Системы query-by-humming: обзор подходов и схема платформы для экспериментов // Труды СПИИРАН. Вып. 7. — СПб.: Наука, 2009.*

**Аннотация.** В работе даётся обзор существующих систем query-by-humming (поиска музыкальных композиций по напетому пользователем фрагменту), использующих извлечение признаков. Кроме того, даётся полное описание наиболее популярных методов извлечения признаков (CENS и MFCC), а также предлагается схема системы для проведения экспериментов, направленных на сравнение разных наборов признаков, методов построения словаря, индексации и поиска.. — Библиограф. 30 назв.

UDC 004.89

*Baltiysky I. A., Nikolenko S. I. Query-by-humming systems: a survey and an experimental setting // SPIIRAS Proceedings. Issue 7. — SPb.: Nauka, 2009.*

**Abstract.** We give a survey of existing query-by-humming information retrieval systems that use feature extraction. The paper contains a complete description of the most popular feature extraction techniques (CENS and MFCC) and offers a design of an experimental system aimed at comparing different feature sets, vocabulary selection, indexing, and search methods.. — Bibl. 30 items.

---

## 1. Введение и постановка задачи

В последнее время получила широкое распространение задача поиска музыки по голосу пользователя (query-by-humming, QBH) [11]. Типичная постановка задачи выглядит следующим образом: имеется набор музыкальных композиций (как правило, достаточно большого объёма), который в дальнейшем будем называть *базой*. Выполняющий поиск человек (будем называть его *пользователь*) знает (слышал) мелодию некоторой части одной из композиций базы и хочет по навязчивой мелодии, звучащей у него в голове, найти настоящую композицию. Пользователь может захотеть найти все исполнения этой мелодии независимо от стиля и исполнителя; кроме того, собственно мелодия может указывать сразу на несколько разных композиций – разных её интерпретаций.

Для поиска пользователь записывает при помощи любого доступного средства записи (как правило, бытового микрофона) собственное исполнение этой мелодии – либо насвистывая её, либо напевая в соответствии с его представлением о ней. Запись этого исполнения будем называть *запросом* (query). От пользователя не должно требоваться специальных вокальных навыков, то есть запрос, скорее всего, будет изначально весьма нечётким, испорченным и лишь условно верным – и это не говоря о шумах, которые наложатся на него по пути к системе.

Задача поиска музыки по голосу ставится так: по запросу необходимо выдать из базы все исполнения композиции, содержащей эту мелодию. В этой работе нас будет интересовать, прежде всего, качество поиска и влияние на него

выбора различных признаков (векторов чисел, в которые при предобработке преобразуются короткие участки композиций). Здесь не будут рассмотрены вопросы эффективности реализации этих признаков в практических системах.

Это позволит нам рассматривать достаточно широкий спектр алгоритмов, лучшие из которых могут указать направление, в котором следует разрабатывать эффективные алгоритмы, применимые в реальных задачах. Критерием качества поиска, таким образом, мы выбираем успешность процедуры поиска независимо от времени её работы.

Мы также покажем, как на основе типичного подхода к решению задачи QBN на основе признаков создать платформу для сравнения различных признаков в решении задачи QBN. Насколько нам известно, это первый систематический научный обзор методов query-by-humming на русском языке.

## 2. Два подхода к query-by-humming

Подходы к решению этой задачи в целом можно разделить на два больших класса [26]. К первому классу относятся методы, использующие для поиска представление музыки в виде некоторой нотации, например, в виде MIDI [7] или признаков, тесно связанных с такого рода нотацией. К таким признакам относятся, например, признаки onset-pitch-duration: из аудиоданных извлекаются высота ноты (pitch) и промежуток времени, в течение которого эта нота звучит (onset — начало звучания, duration – продолжительность) [3]. Фактически эти признаки играют ту же роль, что и запись в MIDI-формате. Некоторые исследователи отказываются и от пауз, утверждая, что пользователь слишком плохо их воспроизводит. Так, например, в [29] база представлена в виде коллекции мелодий, заданных как последовательность пар

$$(N_1, d_1), (N_2, d_2), \dots, (N_k, d_k),$$

где  $N_j$  — нота, а  $d_j$  — её длительность.

К этому классу относится большая часть исследований задачи QBN (см., например, [3, 7, 17, 21, 25, 26, 29]). К сожалению, такой подход накладывает дополнительные ограничения либо на музыкальную базу, либо на сам запрос.

С одной стороны, в символическом виде должна быть представлена сама база. Такое представление может быть получено либо при помощи человека, записывающего мелодии в некотором формате (MIDI, код Парсонса [20] и др.), либо при помощи транскрибирующих алгоритмов. Несмотря на разнообразие таких методов, они, как правило, обладают существенными недостатками либо требуют наличия дополнительной информации (вплоть до участия работы человека). Самая сложная и неоднозначная задача на этом пути – выделение отдельных нот из аудио. Поскольку все существующие решения подвержены различного рода ошибкам, алгоритмы поиска дополнительно усложняются для того, чтобы учесть погрешность, которую внесли сами методы создания символического представления.

С другой стороны, разбор запроса также представляет собой существенную трудность. На данный момент авторам не известен ни один алгоритм, который был бы способен надёжно выделить отдельные ноты из запроса для любого стиля напевания (подобной точки зрения придерживается, например, [21]; см. также [27], где сравниваются разные методы выделения нот). Эту проблему пытаются решить за счёт требований к запросу: так, пользователя просят петь более отрывисто либо петь специальным образом («та-да-да», «фа-фа-фа» и т.п.), как в [21, 25, 27]. Помимо того, что это ограничение может казаться искус-

ственным для пользователя, оно, вообще говоря, иногда невыполнимо: во-первых, в мелодии может быть легато, а во-вторых, иногда дополнительное введение легато упрощает исполнение, что очень важно для пользователя с небогатым опытом пения.

Более того, большинство подходов рассматривают представление базы в виде гомофонических мелодий (в каждый момент времени может звучать только одна нота). Это до некоторой степени оправдано тем, что запрос действительно представляет собой гомофоническую мелодию. Однако это может привести к ухудшению качества поиска. Существуют композиции, в которых одновременно звучат несколько мелодических линий. Пользователь может запомнить и напеть некоторую комбинацию из этих линий, которая не будет совпадать ни с одной из мелодических линий, извлечённых для хранения в базе.

Отдельно отметим работы, посвящённые разложению полифонических мелодий на комбинации гомофонических, то есть, проще говоря, на отдельные ноты. Этот подход, как правило, основан на представлении музыки в виде OPD-признаков (onset, pitch, duration), вычисляющих начало звучания, высоту и продолжительность каждой ноты и производные от этого признаки [22]. Если при таком представлении выделить  $n$ -граммы, описывающие короткие последовательности нот, то для поиска можно будет использовать обычные алгоритмы поиска похожих строк. Поиск музыки при таком подходе фактически становится текстовым; для гомофонической музыки можно использовать методы обычного текстового поиска, а для полифонической музыки – хорошо известные алгоритмы сравнения строк (string matching) [6, 9, 10, 13].

Несмотря на теоретическую привлекательность, эти признаки излишне точны и в большой степени подвержены влиянию ошибок, вносимых процедурой извлечения признаков, и неточностей в воспроизведении мелодии пользователем; в последнее время начали появляться работы, пытающиеся расширить этот набор признаков дополнительной музыкальной информацией [24].

В свете вышеизложенного более красивыми и потенциально более продуктивными подходами представляются подходы, принадлежащие ко второму классу. К ним относятся те методы решения задачи QBN, которые используют *признаки* (features), полученные непосредственно из аудиосигнала, представляющего композиции базы или запрос пользователя. Задача учёта недостатков воспроизведения мелодии пользователем переносится на этап вычисления признаков, что позволяет упростить алгоритмы поиска. Из этого также следует, что признаки должны быть достаточно грубыми. С одной стороны, это должно снизить количество ложных несовпадений, то есть случаев, когда подходящая мелодия отбрасывается. С другой стороны, это избавляет поиск от тех ошибок, которые вносятся в рассмотренных выше подходах алгоритмами отслеживания высоты звука и алгоритмами разбиения на ноты. Учитывая всё вышеизложенное, мы решили посвятить обзор именно алгоритмам поиска музыки по голосу, принадлежащим второй категории.

Авторам неизвестны работы, посвящённые систематическому изучению влияния избранных для построения базы признаков на поведение рассматриваемых алгоритмов поиска. Целью этой и последующих работ является устранение этого пробела; настоящая работа предлагает обзор существующих QBN-систем, основанных на извлечении признаков (feature extraction), и ставит задачу сравнения.

### 3. Общее устройство QВН-систем, основанных на извлечении признаков

В этом разделе мы рассмотрим схему решения задачи QВН, следуя принципам, изложенным в работе Ф. Курта и М. Мюллера [15]. В отличие от авторов, проводивших рассуждения только для специального вида признаков (CENS), мы обобщим изложение на случай произвольных признаков.

Рассмотрим, какие шаги проводятся для поиска очередного запроса  $Q$  в базе  $D$ . С самого начала заметим, что многие из этих шагов идентичны для запроса и базы; однако в то же время из дальнейшего будет очевидно, что действия над базой производятся независимо от действий над запросом и должны быть совершены один раз, до начала приёма запросов. В дальнейшем с учётом этой оговорки все действия над базой и запросом будут для удобства описываться так, как если бы они проводились одновременно. Итак, обработка запроса в базе состоит из следующих этапов.

#### 3.1. Предобработка базы и запроса

Вычисляется последовательность векторов признаков для базы  $(w_0, w_1, \dots, w_{M-1})$ . Вычисляется последовательность векторов признаков для запроса  $(v_0, v_1, \dots, v_{N-1})$ . Каждый из векторов  $v_i, w_j$  лежит в пространстве  $\mathbb{R}^L$ .

Будем обозначать функцию расстояния между векторами через  $\rho: \mathbb{R}^L \times \mathbb{R}^L \rightarrow \mathbb{R}$ . Она должна удовлетворять всем свойствам, которые обычно требуют от функции расстояния (неотрицательность и равенство нулю только для совпадающих элементов, симметричность, неравенство треугольника). Ясно, что выбор функции зависит от характера признаков.

Для улучшения качества поиска (учёта глобальных отклонений запроса в темпе, тональности и т.д.) может составляться несколько запросов (обозначим их число через  $N_Q$ ).

#### 3.2. Поиск соответствия (совпадения)

Сейчас мы рассмотрим, как выполнить такой поиск методом диагонального сопоставления.

*Соответствием (совпадением)* будем называть пару  $(k, N)$ , где  $k$  — позиция соответствия,  $N$  — длина соответствия, то есть запрос  $Q$  *соответствует* последовательности признаков базы  $(w_k, \dots, w_{k+N-1})$ . В методе диагонального сопоставления вычисляется мера расстояния между запросом и фрагментом из базы на позиции  $i$

$$\Delta(i) := \begin{cases} 1 - \frac{1}{N} \sum_{n=0}^{N-1} \rho(w_{i+n}, v_n), & i \in \{0, \dots, M-N\}; \\ \infty, & i \in \{M-N+1, \dots, M-1\}. \end{cases}$$

Ясно, что наилучшим соответствием является пара  $(i_0, N)$  для  $i_0 = \operatorname{argmin}_i \Delta(i)$ . Для поиска второго, третьего и далее лучших соответствий из поиска исключается окрестность  $i_0$ . Название метода становится ясным, если заметить, что приведённое выражение для  $i \in \{0, \dots, M-N\}$  получается путем

суммирования диагональных элементов матрицы  $(\rho(w_m, v_n))_{mn}$ .

Мы рассмотрели метод диагонального сопоставления для одной последовательности векторов признаков запроса. В случае множественных запросов поиск производится следующим образом: минимум ищется как

$$\min(\Delta_1(i), \dots, \Delta_{N_Q}(i)),$$

где каждая из функций расстояния вычисляется отдельно. После того как найден очередной лучший запрос, из области поиска выкидывается окрестность позиции соответствия для всех функций расстояния.

Этого метода могло бы быть достаточно для решения задачи, но он крайне неэффективен (требует операций в количестве, линейном от размера базы) и, что хуже, слишком строг. Поэтому на практике поиск соответствия производится при помощи подхода, основанного на построении и индексировании словаря векторов признаков.

### 3.3. Построение словаря

Обозначим всё множество возможных векторов признаков через  $F$ . Рассмотрим квантовую функцию  $Q: F \rightarrow \{1, \dots, R\}$ :

$$Q(v) = \operatorname{argmin}_{r \in [1:R]} \rho(v, c_r),$$

где  $\rho$  — функция расстояния между признаками, а  $c_r \in C_R = \{c_1, \dots, c_R\}$  — некоторое множество векторов, которое мы будем называть *словарём*. В случае, если  $\operatorname{argmin}$  выдаёт несколько значений, выбирается случайное из них. Таким образом, всё непрерывное множество признаков сводится к конечному набору значений, записанных в словаре.

Выбор словаря  $C_R$  — отдельная задача, отдельный параметр оптимизации системы.

Первый подход к решению этой задачи является универсальным и подходит для различных признаков. Он основан на алгоритме Линде-Бузо-Грея (LBG) [8, 17]. В качестве меры расстояния принимается  $\rho$ . Пусть нам изначально дан некоторый объём словаря  $R$ . Применим итеративный подход. На первом шаге словарь

$$C_R^0 = \{c_1^0, \dots, c_R^0\}$$

выбирается произвольным образом. При переходе на каждый новый шаг  $l$  вычисляется функция ошибки как среднее расстояние между вектором базы и соответствующим ему вектором из словаря, составленного на предыдущем шаге:

$$\varepsilon^l = \frac{1}{M} \rho \left( w_m, c_{Q^l(w_m)}^l \right),$$

где  $Q^l(w)$  — функция квантования по словарю  $C_R^l$ .

Алгоритм останавливается, если ошибка становится меньше порогового значения. В противном случае вычисляется новый словарь: для каждого кластера, составленного на предыдущем шаге, создается новый вектор, равный центру масс старого кластера. Обозначим через  $P_r^l = (Q^l)^{-1}(r)$  множество признаков, отнесённых квантовой функцией  $Q^l$  к классу  $r$ . Тогда все векторы

словаря пересчитываются следующим образом:

$$c_r^{l+1} = \frac{1}{|P_r^l|} \sum_{v \in P_r^l} v.$$

Если признаки нормализованы (длина каждого вектора равна 1 или другой константе), то на этом же этапе полученный центр масс нужно спроецировать на сферу соответствующей размерности (центр масс нескольких точек единичной сферы на ней, скорее всего, не лежит).

Другой возможный подход — применение априорного знания о полученных векторах признаков. В некоторых случаях можно составить универсальный словарь, базируясь на специальных свойствах тех или иных признаков. Это возможно, например, для признаков CENS, о чём мы подробнее расскажем ниже.

Первый из этих подходов хорош тем, что позволяет выбирать словарь произвольного размера, в то время как размер словаря во втором методе диктуется особенностями признаков и поэтому фиксирован. В то же время, первый метод требует базы для обучения, а второй метод может дать предположительно лучший результат, если специальные свойства достаточно хорошо разделяют признаки. Следовательно, в тех случаях, когда возможен второй подход, нужно использовать оба с тем, чтобы сравнить их между собой.

### 3.4. Индексирование и поиск точного соответствия

После составления словаря база представляется в квантованном по словарю виде  $Q[W] = (r_0, r_1, \dots, r_{M-1}), r_m = Q(w_m)$ . Затем строится инвертированный индекс

$$L(r) = \{m \in \{0, \dots, M-1\} \mid r_m = r\}, r \in \{1, \dots, R\}.$$

Квантуется и запрос — полученную последовательность обозначим  $(s_0, s_1, \dots, s_{N-1})$ .

Назовём точным соответствием ситуацию, при которой последовательность  $Q[V]$  и подпоследовательность  $Q[W]$  той же длины совпадают полностью. Множество всех точных совпадений можно вычислить как

$$\bigcap_{0 \leq n < N} (L(s_n) - n),$$

где операция вычитания применяется поэлементно. Корректность этой формулы понять несложно: на позиции  $k$  имеется соответствие, если и только если  $\forall n \in \{0, \dots, N-1\}$  вектор  $s_n$  находится на позиции  $k+n$  в  $Q[W]$ , то есть  $k+n \in L(s_n)$ , но это и значит, что  $k \in L(s_n) - n$ .

Однако поиск точных соответствий слишком строг и почти наверняка приведёт к большому количеству ложных несовпадений (искомый запрос не попадёт в список лучших). Ослабим условие соответствия, применяя вместе следующие два (независимых) подхода: *нечёткое сравнение* и *поиск, устойчивый к ошибкам*.

### 3.5. Нечёткое сравнение

Квантующая функция, рассмотренная выше, на самом деле достаточно «опасна». Если вектор находится на границе кластеров, достаточно совсем небольшой ошибки, чтобы значение квантующей функции изменилось. Поэтому нужно изменить определение соответствия или определение квантованного вектора признаков так, чтобы исправить этот недостаток.

Рассмотрим вместо квантующей функции многозначное отношение, сопоставляющее вектору признаков не одно значение (класс), как раньше, а множество, состоящее из этого значения и нескольких соседних, т.е.

$$Q_{\lambda}^p(v) = \{r \mid \rho(v, c_r) \leq P\}, |Q_{\lambda}^p| \leq \lambda.$$

Новое отношение применяется ко всем векторам запроса  $V$ :

$$Q_{\lambda}^p[V] = (S_0, S_1, \dots, S_{N-1}).$$

Будем говорить, что пара  $(k, N)$  является нечётким соответствием, если  $\forall n \in \{0, \dots, N-1\} r_{k+n} \in S_n$ . Обратные списки теперь выглядят так:

$$L(S_n) = \bigcup_{s \in S_n} L(s).$$

Вычисление множества нечётких совпадений производится так же, как в случае точных соответствий.

### 3.6. Устойчивый к ошибкам поиск

Идея проста: разрешим не более  $\nu$  ошибок в соответствиях. Формально говоря, введём функцию  $\mu: \{0, \dots, M-1\} \rightarrow \{0, \dots, N\}$ :

$$\mu(m) := \left| \left\{ n \in \{0, \dots, N-1\} \mid m \in (L(S_n) - n) \right\} \right|$$

и будем говорить, что  $(k, N)$  — *нечёткое соответствие с не более чем  $\nu$  ошибками*, если  $\mu(k) \geq N - \nu$ . Подсчёт этой функции производится либо при помощи динамического программирования [5], либо при помощи хеширования [28].

### 3.7. Ранжирование

Искомая мелодия может встречаться в различной аранжировке в базе, равно как и встречаться несколько раз в одной композиции. Могут быть найдены похожие в некотором смысле мелодии, возможно, далёкие от того, что предполагал пользователь. Это приводит к необходимости научиться вводить ранжирование результатов.

Рассмотрим двухэтапную процедуру ранжирования. Первый этап предполагает оценку результатов по количеству ошибок, разрешённых при отказоустойчивом поиске. Вспомним, что на самом деле в поиске участвует несколько запросов с различными длинами, и отнормируем эту оценку по длине найденного соответствия:

$$\Theta(k, N) := \frac{\mu(k)}{N} \in [0, 1].$$

Это, однако, достаточно грубый инструмент оценки результатов поиска, поэтому мы им воспользуемся только для того, чтобы отсеять заведомо нерелевантные результаты. Для этого можно взять только несколько лучших результатов или просто некоторое априорно заданное их число.

Второй этап использует уже рассматривавшийся выше метод диагонального сопоставления для того, чтобы вычислить функции расстояния для небольшой окрестности лучших соответствий, полученных после первого этапа. Окончательное ранжирование производится по полученной функции расстояния.

### 3.8. Общая схема QВН-системы на базе извлечения признаков

В качестве итога этого раздела попробуем коротко сформулировать основные принципы построения QВН-системы на базе извлечения признаков.

1. Признаки квантуются некоторым словарём, то есть вместо векторов признаков рассматриваются последовательности символов из относительно небольшого (до 1000 символов) словаря.

2. Для улучшения поиска из одного запроса могут составляться несколько запросов для учета глобальных темпоральных и тональных отклонений.

3. Индексация производится по методу обратных списков. Сравнение можно эффективно реализовать на его основе.

4. Для устойчивости к ошибкам применяется нечёткое квантование и сравнение по принадлежности, а не по равенству.

5. Разрешается некоторое количество ошибок при сравнении.

6. Выводятся только соответствия, относительное количество ошибок для которых меньше порогового; они ранжируются по функции расстояния, полученной методом диагонального соответствия.

Два следующих раздела будут посвящены признакам, которые могут применяться для решения задачи QВН вышеописанным способом.

## 4. CENS-признаки

Вычисление признаков CENS (Chroma Energy Normalized Statistics) основывается на идее рассматривать в качестве главного аспекта музыки интонационный аспект, то есть мелодию и гармонию. Восприятие интонационного аспекта тесно связано с возможностью человеческой слуховой системы определять высоту звука. Поскольку построение CENS-признаков использует теорию восприятия высоты тона и практические следствия из неё в музыкальной культуре, начнём этот раздел с некоторых сведений из психоакустики и теории музыки.

Что такое высота звука? Широко распространено определение, данное в стандарте ANSI-1994: «Высота (Pitch) – это атрибут слухового ощущения в терминах, в которых звуки можно расположить по шкале от низких к высоким. Высота зависит главным образом от частоты звукового стимула, но она также зависит от звукового давления и от формы волны». Подобного определения придерживается и Беранек [2].

Пара высот образует *интервал*. Тип интервала зависит от отношения частот, соответствующих его высотам. Ключевое значение в музыке играет интервал между высотами, частоты которых соотносятся как 2 : 1; он называется *октавой*.

Начиная с XVII-XVIII веков, основная часть западноевропейской музыки строится на основе *равномерно темперированного строя*<sup>1</sup>. В дальнейшем мы будем рассматривать только музыку, использующую этот строй. Это ограничение достаточно разумно, поскольку практически вся популярная и подавляющее большинство академической музыки удовлетворяет этому требованию.

Использование равномерно темперированного строя (далее РТС) в некоторой композиции означает следующее: можно считать, что частоты всех высот, встречающихся в этой композиции, содержатся в конечном наборе частот,

<sup>1</sup> Самым известным аргументом в пользу этого строя был, конечно, «Хорошо темперированный клавир» И. С. Баха, но на самом деле равномерно темперированный строй использовал и иллюстрировал своими произведениями ещё Винченцо Галилей, отец Галилео.



определяемых этим строем. Обозначим такой набор  $F_{n\text{-EDO}}$  — это набор частот для РТС с  $n$  полутонами ( $n\text{-EDO}$  означает Equal Division of Octave – равномерное деление октавы). Как следует из названия, набор получается путём (логарифмического) деления октавы на  $n$  равных частей.

Существуют строи, основанные на делении других интервалов, но мы не будем их рассматривать. Как правило,  $n = 12$  (классический РТС, применяется практически повсеместно в современной популярной музыке), также существуют (хотя редко используются) строи 24-EDO, 19-EDO, 31-EDO [4]. Мы ограничим наше рассмотрение случаем 12-EDO.

Математически построение набора  $F_{n\text{-EDO}}$  можно описать так. Выбирается некоторая основная частота  $f_{\text{base}}$ , ей задаётся некоторый номер  $\rho_{\text{base}}$ . Затем определяется диапазон номеров частот, которые будут входить в набор:  $[\rho_{\text{min}}, \rho_{\text{max}}]$ . Тогда набор частот определяется так:

$$F_{n\text{-EDO}} = \left\{ f_p : f_p = f_{\text{base}} 2^{\frac{\rho - \rho_{\text{base}}}{n}}, \rho = \rho_{\text{min}}, \rho_{\text{max}} \right\}.$$

Начиная с середины XX века, принято брать в качестве  $f_{\text{base}}$  частоту 440 Гц. В выборе остальных констант мы будем следовать Мюллеру [18]:  $\rho_{\text{base}} = 69$ ,  $\rho_{\text{min}} = 21$ ,  $\rho_{\text{max}} = 108$ .

Обратим внимание на то, что все ноты для построенного набора разбиваются на  $n = 12$  классов. Их называют *классами высот* (pitch classes). Для музыки они важны тем, что ноты из одного класса играют одну и ту же роль в восприятии гармонии (это называется *октавной эквивалентностью*). Мы будем также пользоваться и другим названием классов — *хромы* (от chroma, «цвет»).

Октавная эквивалентность отражена и в системе названий для высот. По традиции, каждая хрома имеет свое имя (например: ля, соль-диез или, что в РТС 12-EDO то же самое, ля-бемоль). Нота задаётся номером октавы и названием хромы: так, базовую ноту, соответствующую по современным стандартам частоте 440 Гц, принято называть «ля первой октавы».

Учитывая введённое понятие хром, перейдём к описанию CENS-признаков. Кратко говоря, каждый вектор CENS-признаков является нормализованным распределением энергии по хромам на небольшом промежутке времени. Изложим более подробно построение, которое состоит из нескольких этапов.

#### 4.1. Выделение поддиапазонов (subbands)

На этом этапе из исходного сигнала  $x$  выделяются поддиапазоны, соответствующие каждой частоте из набора  $F_{12\text{-EDO}}$ . Это делают путём фильтрации  $x$  полосовыми фильтрами с центральными частотами  $f_p \in F_{12\text{-EDO}}$ . Формально эту операцию можно записать так:  $x_0 \rightarrow \left\{ x_p^F \right\}_{\rho=\rho_{\text{min}}}^{\rho_{\text{max}}}$ , где  $x_p^F$  — сигнал, полученный на выходе фильтра с центральной частотой  $f_p$ .

Опишем спецификацию фильтров. Для всех фильтров вводится общая добротность  $Q = 25$ . Полоса пропускания выбирается так, чтобы её центральной частотой была  $f_p$ . Ширина полосы пропускания определяется как  $w = \frac{f_p}{Q}$ .

Центральная частота располагается в центре полосы пропускания. Таким образом, полоса пропускания ограничена частотами  $f_{\text{pass},1,2} = f_p \pm \frac{W}{2}$ . Частоты полос подавления вычисляются так:  $f_{\text{stop},1,2} = f_p \pm W$ . Неравномерность в полосе пропускания должна составлять 1Дб, подавление должно составлять 50 Дб [18].

Поскольку при проведении экспериментов необходимо обрабатывать большие объёмы данных (по меньшей мере порядка десятков гигабайт), авторам потребовалось добиться разумного быстродействия при реализации этого этапа. Это привело к необходимости, во-первых, ускорить фильтрацию — для этого необходимо минимизировать порядок фильтра, — а во-вторых, сократить объём обрабатываемых данных за счёт понижения частоты дискретизации.

#### 4.1.1. Фильтрация.

Для фильтрации на этом этапе естественно использовать фильтры с бесконечной импульсной характеристикой (БИХ-фильтры), поскольку для данной спецификации они имеют существенно меньший порядок, чем фильтры с конечно-импульсной характеристикой (КИХ-фильтры) [23]. Один из распространённых подходов к созданию цифровых фильтров по заданной спецификации заключается в расчёте аналогового фильтра с последующим преобразованием его в цифровой фильтр. Говоря подробнее, по изначальной спецификации для полосового фильтра необходимо составить спецификацию для аналогового НЧ-фильтра, а затем, пользуясь известными формулами для аналоговых фильтров, найти его корни и полюса. К полученным точкам применяется билинейное преобразование, переводящее эти точки в корни и полюса соответствующего цифрового полосового фильтра [23].

Исходная спецификация для полосового фильтра была приведена выше. Для аналогового НЧ-фильтра необходимо преобразовать эту спецификацию в значения частот пропускания и подавления  $\Omega_p, \Omega_s$ . Требования к неравномерности в полосе пропускания и подавлению остаются теми же, хотя для аналоговых фильтров их иногда выражают как относительные отклонения (ripples):

$$\varepsilon_p = \sqrt{10^{A_p/10} - 1}, \quad \varepsilon_s = \sqrt{10^{A_s/10} - 1}.$$

Также вводят коэффициенты избирательности  $k = \frac{\Omega_p}{\Omega_s}$  и  $k_1 = \frac{\varepsilon_p}{\varepsilon_s}$ .

В качестве аналогового прототипа был выбран эллиптический фильтр. Модуль импульсной характеристики выглядит следующим образом:

$$|H(\Omega)|^2 = \frac{1}{1 + \varepsilon_p^2 \text{cd}^2(NuK_1, k_1)}, \quad \text{где } K_1 = \int_0^{\pi/2} \frac{d\theta}{\sqrt{1 - k_1^2 \sin^2 \theta}}.$$

Здесь  $N$  — порядок фильтра,  $K_1 = K(k_1)$  — полный эллиптический интеграл первого рода с параметром  $k_1$ ,  $\text{cd}$  — эллиптическая функция Якоби, а  $u$  таково, что  $\frac{\Omega}{\Omega_p} = \text{cd}(uK, k)$  для  $K = K(k)$  [19].

Несмотря на столь хорошее поведение модуля импульсной характеристики, эллиптические фильтры имеют существенно нелинейную фазовую характеристику. Линейность фазовой характеристики очень важна. Во-первых, нелинейность фазы приводит к искажениям сигнала. Однако даже если бы в преде-

лах полосы пропускания фазу  $\arg H(\Omega)$  можно было приблизительно считать линейной, то для фильтров с различными центральными частотами групповая задержка отфильтрованного сигнала была бы различной. Это может привести, например, к тому, что нескольким одновременно звучащим нотам будут соответствовать различные участки отфильтрованных сигналов, то есть вместо аккорда получится набор из нескольких различных нот.

Для обеспечения линейности фазовой характеристики хорошо подходит метод, применяемый и для остальных БИХ-фильтров — так называемый метод forward-backward filtering. Он заключается в том, что некоторый фильтр  $G$  применяется для исходного сигнала, затем результирующий сигнал обращается во времени, к нему снова применяется  $G$ , и полученный сигнал снова обращается во времени:

$$y_f(t) = Gx_0, y_{fb}^R(t) = Gy_f^R,$$

где  $f$  означает прямую фильтрацию,  $b$  — обратную фильтрацию, а  $R$  — оператор обращения во времени [12].

Выбор эллиптического фильтра обоснован тем, что он имеет наименьший порядок среди всех фильтров для данного набора спецификаций [23].

Поиск корней и полюсов эллиптического фильтра, а также процедура построения цифрового фильтра, подробно описаны в [19]. Для реализации необходимы функции Якоби  $cd$  и  $sn$ , а также функция  $sn^{-1}$ , в некотором смысле обратная к  $sn$  (она сопоставляет  $w$  и  $k$  такое  $u$ , что  $w = sn(uK(k), k)$ ). Авторам не удалось найти в свободном доступе библиотек, в которых эти функции были бы реализованы, поэтому была написана собственная реализация на основе [1, 19].

#### 4.1.2. Минимизация объёма вычислений путем понижения частоты дискретизации.

Поскольку описанная выше процедура фильтрации выделяет из сигнала только те частоты, которые попадают в полосу пропускания каждого из фильтров, нас не будут интересовать те частоты, которые находятся выше верхней частоты пропускания фильтра с самой большой центральной частотой  $f_{p_{\max}}$ .

Типичная частота дискретизации аудио составляет 44100 Гц (например, формат CDDA), что позволяет представлять частоты до 22050 Гц (частота Найквиста [23]).

В то же время, при  $p_{\max} = 108$   $f_{108} = 440 \cdot 2^{12} = 4186.009$ . Видно, что можно значительно сократить частоту дискретизации — до  $2f_{108} = 8372.018$ . Разумеется, при этом необходимо ограничивать спектр сигнала НЧ-фильтром для избежания наложения спектров (aliasing). Развивая эту мысль, мы приходим к идее банка фильтров с несколькими частотами дискретизации (multirate filter bank).

Идея заключается в том, чтобы разбить весь диапазон частот фильтр-банка на подгруппы, в каждой из которых сигнал представлен с общей для этой группы частотой дискретизации. При выборе количества групп и значений частоты дискретизации нужно учитывать следующие факторы.

- (а) Понижение частоты дискретизации в целое число раз  $n$  можно реализовать очень эффективно при помощи прореживания (decimation) — достаточно оставить в сигнале только сэмплы, стоящие на позициях

$kn, k \in \mathbb{Z}_{+0}$ . Из этого следует правило выбора частот дискретизации: они должны в целое число раз отличаться от исходной частоты.

- (б) Для НЧ-фильтрации сигнала перед прореживанием должен применяться линейный фильтр с высоким подавлением. Как правило, для этого используется БИХ-фильтр. Как правило, БИХ-фильтры, удовлетворяющие условиям высокого подавления, имеют высокие порядки (1000 и более), что делает понижение частоты дискретизации достаточно трудоёмкой операцией. Это накладывает ограничение сверху на количество подгрупп фильтров.

Исходя из этого, выбирается немного (например, 3) групп с частотами дискретизации в 2, 10 и 20 раз меньше исходной [18]:  $[f_{21}, f_{56}]$ ,  $[f_{57}, f_{92}]$ ,  $[f_{93}, f_{108}]$ . Затем можно двигаться от самой верхней частоты к самой нижней, прореживая исходный сигнал при смене подгруппы.

#### 4.2. Средняя энергия на коротком промежутке времени (STMSP, Short Time Mean Square Power).

На втором этапе для полученных поддиапазонов вычисляется средняя энергия в прямоугольном окне ширины  $w$ :

$$\sum_{k \in \left[ n - \left\lfloor \frac{w}{2} \right\rfloor : n + \left\lfloor \frac{w}{2} \right\rfloor \right]} |x_p^f(k)|^2.$$

Для дальнейшего сокращения объёмов вычислений полученный сигнал прореживается в  $d$  раз. При этом ширина окна  $w$  выбирается как  $2d + 1$ , а  $d$  рассчитывается, исходя из требуемой частоты векторов признаков  $F_{\text{STMSP}}$ . При этом надо учитывать, что полученные на предыдущем шаге поддиапазоны имеют различные частоты дискретизации в зависимости от подгруппы частот.

Этот этап можно символически изобразить так:  $\{x_p^f\} \rightarrow \{x_p^{fs}\}$ , где  $s$  означает операцию взятия средней локальной энергии.

Заметим, что вычисления на этапах вплоть до этого можно проводить параллельно внутри каждой группы, что также даёт возможность ускорить вычисление CENS-признаков за счёт параллелизации.

#### 4.3. Нормализованные хромы (Normalized chroma features).

Полученные на предыдущем этапе распределения локальной энергии по поддиапазонам преобразуются в распределения по хромам:

$$\{x_p^{fs}\} \rightarrow \left\{ y_q : y_q(t) = \sum_{p \equiv q \pmod{12}} x_p^{fs}(t), q = \overline{1, 12} \right\} \rightarrow \{y_q^N\}, \text{ где}$$

$$y_q^N(t) = \begin{cases} \frac{y_q(t)}{\|y(t)\|_1}, & \|y(t)\|_1 > 0 \\ 0, & \|y(t)\|_1 = 0 \end{cases} \text{ для } \|y(t)\|_1 = \sum_{r=1}^{12} |y_r(t)|.$$

#### 4.4. Квантование.

При квантовании осуществляется переход  $\{y_q^{N_1}\} \rightarrow \{z_q\}$ , где

$$z_q(t) = \tau(y_q^N(t)), \tau(a) := \begin{cases} 0, & 0 \leq a < 0,05 \\ 1, & 0,05 \leq a < 0,1 \\ 2, & 0,1 \leq a < 0,2 \\ 3, & 0,2 \leq a < 0,4 \\ 4, & 0,4 \leq a \leq 1 \end{cases}.$$

Заметим, что пороги квантования распределены по логарифмическому закону. Это соответствует законам восприятия человеком громкости звука (закон Вебера-Фехнера, формула Стивенса [30]).

#### 4.5. Локальная нормализация.

Последний этап характеризуется парой параметров  $(w, d)$ , имеющих тот же смысл, что и на шаге 2: к сигналам, полученным на предыдущем этапе, применяется окно ширины  $w$ , затем он прореживается в  $d$  раз. Ширину окна выбирают как  $w = 4d + 1$  [15]. Полученные сигналы нормализуются аналогично этапу 3, но по Евклидовой норме:

$$\{z_q\} \rightarrow \{z_q^{(w,d)}\} \rightarrow \{z_q^{(w,d)N} : z_q^{(w,d)N}(t) = \pi(z_q^{(w,d)}(t))\}, \text{ где}$$

$$\pi(a_q) = \begin{cases} \frac{a_q}{\|a_q\|}, \|a_q\| > 0 \\ \frac{(1,1,\dots,1)}{\sqrt{12}}, \|a_q\| = 0 \end{cases}.$$

Последовательность векторов, полученная в результате всех этих шагов, — это и есть CENS-признаки. Мы будем обозначать их через  $CENS_d^w$ .

Заметим, что процедура получения  $CENS_d^w$ -признаков из  $\{z_q\}$  не является трудоёмкой. Это позволяет быстро получать  $CENS_d^w$  для различных  $(w, d)$ . Такой приём используется в поиске аудио: если для базы получены  $CENS_{10}^{41}$ , то для запроса получают  $CENS_{d_j}^{w_j}$  при  $d_j = 7 \dots 14$ , что позволяет учесть отклонения в темпе запроса в пределах  $\pm 40\%$  от темпа соответствующей записи в базе.

## 5. MFCC-признаки

В предыдущем разделе уже упоминалось широко известное определение высоты звука. Как мы отметили, главным образом высота зависит от основной частоты звука. В психоакустике вводится единица измерения высоты чистого тона – мел. За 1000 мел принимается высота (синусоидального) звука с частотой 1000 Гц интенсивностью 60 Дб (измеряемых относительно 0,0002 микробар). Зависимость этой психофизической величины от физической (частоты) устанавливается экспериментально (подробнее см. [2, 14]). Полученная кривая описывается приближённым уравнением  $B(f) = 1125 \ln\left(1 + \frac{f}{700}\right)$ ; ясно, что

$$B^{-1}(f) = 700 \left( e^{\frac{b}{1125}} - 1 \right); \text{ см. также подробное описание в [14].}$$

Введём понятие MFCC, или Mel-Frequency Cepstrum Coefficients (мел-частотные кепстральные коэффициенты). Фактически признаки MFCC являются представлением исходного сигнала  $x$  в виде вещественного кепстра по шкале мелов для взятых в некотором окне участков сигнала, полученного из дискретного Фурье-образа  $X$ . Под вещественным кепстром понимается преобразование

$$c[n] = \frac{1}{2\pi} \int_{-\pi}^{\pi} \ln |X(e^{j\omega})| e^{j\omega n} d\omega$$

(само слово «кепстр» происходит от перевернутого «спектр», указывая на спектральную природу преобразования) [14].

Построение MFCC производится следующим образом. Сначала выбирается число коэффициентов в результирующем представлении  $M$  и диапазон частот, которые будут участвовать в представлении  $[f_L, f_H]$ . Рассмотрим  $M+1$  частот  $f_m$ , равномерно распределённых по мел-шкале в выбранном диапазоне:

$$f_m = \frac{N}{F_s} B^{-1} \left( B(f_L) + m \frac{B(f_H) - B(f_L)}{M+1} \right).$$

Для исходного сигнала  $x_0$ , взятого в некотором окне (например, окне Хэмминга), вычисляется дискретное преобразование Фурье (ДПФ)  $X_0$  размера  $N$ . Для каждой частоты  $f_m$  рассмотрим треугольный фильтр с частотной характеристикой  $H_m[k]$ , где  $k$  — номер ДПФ-ячейки,  $0 \leq k < N$ :

$$H_m[k] = \begin{cases} 0, k < LF(m), \\ \frac{2(k - LF(m))}{(RF(m) - LF(m))(CF(m) - LF(m))}, LF(m) \leq k \leq CF(m), \\ \frac{2(RF(m) - k)}{(RF(m) - LF(m))(RF(m) - CF(m))}, CF(m) \leq k \leq RF(m), \\ 0, k > RF(m), \end{cases}$$

где  $LF(m) = \begin{cases} f_{m-1}, m > 0 \\ f_L, m = 0 \end{cases}$ ,  $CF(m) = f_m$ ,  $RF(m) = f_{m+1}$ .

Рассмотрим логарифмическую энергию:

$$S_m = \ln \left( \sum_{k=0}^{n-1} |X_0[k]|^2 H_m[k] \right), 0 \leq m < M$$

Чтобы вычислить кепстр мел-частоты, применим дискретное косинусоидальное преобразование (DCT-II):

$$c[n] = \sum_{m=0}^{M-1} S_m \cos\left(\pi n \frac{m+1/2}{M}\right).$$

Первые  $M$  значений полученного кепстра и называются мел-частотными кепстральными коэффициентами (MFCC) [14]. Заметим, что первый из этих коэффициентов имеет физический смысл энергии сигнала на взятом промежутке времени.

Для получения признаков на основе MFCC будем брать  $M = 13$ . Вычисления будем производить в окне Хэмминга небольшого размера (200 мс) со сдвигом окна 100 мс [23].

## 6. Описание схемы экспериментов

В этом разделе мы опишем устройство платформы для экспериментов с различными признаками и критерии оценки качества поиска. Схема работы изображена на рис. 1.

Изначально для экспериментов выбирается база  $D$ . Какие требования разумно предъявить для базы?

1. Требования к характеру записей, содержащихся в базе. С одной стороны, записи должны быть разнообразными (разные стили, авторы, исполнители). С другой стороны, в пределах каждого стиля должно быть достаточно большое количество записей (например, для современной популярной музыки — как минимум один альбом на автора). Хорошее представление о разнообразии стилей можно получить, посетив сайты, посвященные музыкальной тематике<sup>2</sup>.

2. Требования к размеру базы. Как правило, такие требования задаются общим временем звучания. Размер должен быть достаточно большим, чтобы удовлетворить предыдущее требование.

3. Все записи в базе должны удовлетворять требованиям, предъявляемым всеми признаками, участвующими в эксперименте. Например, CENS-признаки определены только для музыки, построенной в РТС 12-EDO (хотя, как мы уже замечали выше, практически вся популярная музыка и академическая музыка XVIII-XX веков удовлетворяет этому требованию<sup>3</sup>).

Для экспериментов также подготавливаются запросы  $\{Q_j\}_{j=1}^{T_j}$ . Запросы должны содержать мелодии различных жанров. Для каждого запроса методом экспертной оценки подготавливается список композиций из базы, содержащих этот запрос (т.е. ответы на запросы). Для каждого запроса должна быть хотя бы одна соответствующая композиция в базе. Обозначим эти списки ответов как  $A_j = A(Q_j)$ .

Далее, для базы строятся наборы признаков для всех исследуемых типов признаков (их строители обозначены как  $FE^i$ , от слов «feature extractor»). Для

<sup>2</sup> [www.last.fm](http://www.last.fm), [www.allmusic.com](http://www.allmusic.com) и другие.

<sup>3</sup> С двадцатым веком, правда, ситуация чуть более интересная; в XX в. появилась возможность конструировать новые инструменты, на которых можно было получать более чистую гармонию, чем классический равномерно темперированный строй с двенадцатью ступенями октавы. Например, советский изобретатель Евгений Мурзин создал синтезатор АНС (название – в честь Скрябина), в октаве которого не 12, а 72 ступени. Но, конечно, подавляющее большинство музыкальных композиций всё равно создаётся в РТС 12-EDO.

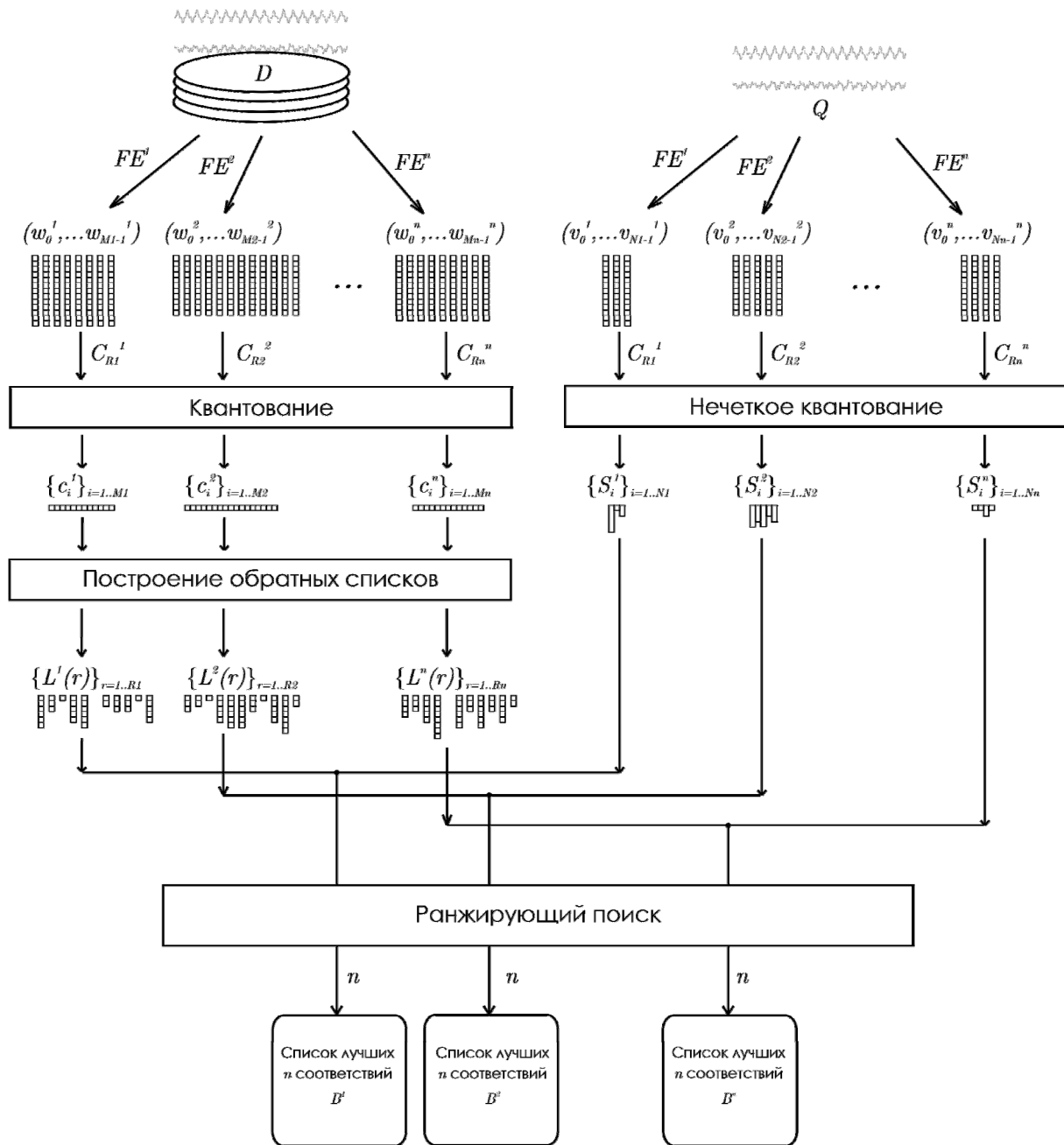


Рис. 1. Общее устройство экспериментальной платформы.

них выбираются словари  $C_{R^i}^i$ , причём те типы признаков, для которых возможны несколько вариантов выбора словаря, участвуют дважды с различными словарями. Также возможен вариант, когда один и тот же тип признаков участвует несколько раз с различными значениями  $R^i$ . Полученные признаки квантуются по словарям  $C_{R^i}^i$ , по ним строятся инвертированные индексы  $L^i(r), r = 1..R^i$  (см. рис. 1).

Затем для каждого запроса  $Q_j$  запускается алгоритм поиска (см. рис. 1). Заметим, что для улучшения качества поиска из каждого запроса может быть произведено несколько; при этом засчитывается общий результат эксперимента. После извлечения признаков и нечёткого квантования алгоритм ранжирую-



щего поиска выдаёт для каждого из исследуемых признаков ограниченный первыми  $n$  лучшими результатами список ответов  $B_{nj}^i$  на запрос  $Q_j$ .

Эксперимент повторяется для нескольких выбранных значений  $n$ . Оценка результатов производится отдельно для каждого  $n$  следующим образом: качество поиска для каждого запроса  $Q_j$  определяется как

$$\kappa_{nij} = \frac{|A_j \cap B_{nj}^i|}{|A_j|}.$$

Сравнение качества работы признаков будем производить для всех  $n$  по среднему по запросам качеству поиска:

$$\kappa_{ni} = \frac{\sum_j \kappa_{nij}}{T_j}.$$

## 7. Заключение и открытые вопросы

В этой работе мы провели обзор существующих методов построения систем query-by-humming, основанных на извлечении признаков, подробно описали два наиболее известных и популярных набора признаков, а затем описали схему экспериментов, направленных на сравнительный анализ разных подходов к вычислению признаков, построению словарей, индексированию и поиску.

Наша ближайшая работа в этом направлении будет носить экспериментальный характер. Мы планируем провести эксперименты по указанной схеме, сравнив разные наборы признаков (CENS, MFCC, их модификации, а также другие признаки) с использованием разных методов построения словарей и индексации. Статья подготовлена при поддержке РФФИ (проекты 08-01-00640-а, 09-01-00784-а).

## Литература

1. *Abramowitz M., Stegun I. A.* Handbook of Mathematical Functions. Dover Publications, 1972.
2. *Beraneck L. L.* Acoustics. Acoustical Society of America, 1993.
3. *Bosma M., Veltkamp R. C., Wiering F.* Muugle: A framework for the comparison of Music Information Retrieval methods // Proceedings of the ICMPC. 2006. P. 1297–1303.
4. *Burns E. M.* Intervals, Scales, and Tuning // The Psychology of Music / Ed. by D. Deutsch. Academic Press, 1999. P. 215–264.
5. *Clausen M., Kurth F.* A Unified Approach to Content-Based and Fault Tolerant Music Identification // CW '02: Proceedings of the First International Symposium on Cyber Worlds (CW'02). Washington, DC, USA: IEEE Computer Society, 2002. P. 56.
6. *Crawford T., Iliopoulos C. S., Raman R.* String-Matching Techniques for Musical Similarity and Melodic Recognition // Computing in Musicology. 1998. Vol. 11. P. 73–100.
7. *Dannenbergh R. B., Birmingham W. P., Tzanetakis G. P., Meek C. P., Hu N. P., Pardo B. P.* The MUSART Testbed for Query-by-Humming Evaluation // Computer Music Journal. 2004. Vol. 28, N. 2. P. 34–48.
8. *Deller J. R., Proakis J. G., Hansen J. H. L.* Discrete-time processing of speech signals. IEEE Computer Society Press, 1999.
9. *Doraisamy S., Ruger S.* Robust Polyphonic Music Retrieval with N-grams // Journal of Intelligent Information Systems. 2003. Vol. 21, N. 1. P. 53–70.
10. *Downie J. S.* Evaluating a Simple Approach to Music Information Retrieval: Conceiving Melodic N-

- Grams as Text: Ph.D. thesis / University of Western Ontario. Canada, 1999.
11. *Ghias A., Logan J., Chamberlin D., Smith B. C.* Query by humming: musical information retrieval in an audio database // MULTIMEDIA '95: Proceedings of the third ACM international conference on Multimedia. New York, NY, USA: ACM, 1995. P. 231–236.
  12. *Gustafsson F.* Determining the initial states in forward-backward filtering // IEEE Transactions on Signal Processing. 1996. Vol. 46. P. 988–992.
  13. *Hanna P., Robine M., Ferraro P., Allali J.* Improvements of Alignment Algorithms for Polyphonic Music Retrieval // Computer Music Modeling and Retrieval 2008. Copenhagen, Denmark: 2008. P. 244–251.
  14. *Huang X., Acero A., Hon H.-W.* Spoken Language Processing: a guide to theory, algorithm, and system development. Prentice Hall, 2001. 980 p.
  15. *Kurth F., Muller M.* Efficient Index-Based Audio Matching // IEEE Transactions on Audio, Speech, and Language Processing. February 2008. Vol. 16, N. 2. P. 382–395.
  16. *Lau E., Ding A., Calvin, J.* MusicDB: A Query by Humming System. Final Project Report, Massachusetts Institute of Technology. 2005.
  17. *Linde Y., Buzo A., Gray R.* An Algorithm for Vector Quantizer Design // IEEE Transactions on Communications. 1980. Vol. 28. P. 84–94.
  18. *Muller M.* Information Retrieval for Music and Motion. Springer-Verlag, 2007.
  19. *Orfanidis S. J.* Lecture notes on elliptic filter design. <http://www.ece.rutgers.edu/orfanidi/ece521/>. 2007.
  20. *Parsons D.* The Directory of Tunes and Musical Themes. S. Brown, 1975.
  21. *Pauws S.* CubyHum: a fully operational "query by humming" system // Proceedings of the 3rd International Conference on Music Information Retrieval (ISMIR'02). 2002.
  22. *Pickens J., Crawford T.* Harmonic models for polyphonic music retrieval // CIKM '02: Proceedings of the eleventh international conference on Information and knowledge management. New York, NY, USA: ACM, 2002. P. 430–437.
  23. *Proakis J. G., Manolakis D. G.* Digital Signal Processing: Principles, Algorithms, and Applications. Prentice Hall, 1995.
  24. *Robine M., Hanna P., Ferraro P.* Music Similarity: Improvements of Edit-based Algorithms by Considering Music Theory // Proceedings of the ACM SIGMM International Workshop on Multimedia Information Retrieval (MIR). Augsburg, Germany: 2007. P. 135–141.
  25. *Typke R.* Music Retrieval based on Melodic Similarity. Rainer Typke, 2007.
  26. *Typke R., Wiering F., Veltkamp R. C.* A survey on music information retrieval systems // Proceedings of the International Conference on Music Information Retrieval. 2005. P. 153–160.
  27. *Uitdenbogerd A. L., Zobel J.* Manipulation of music for melody matching // MULTIMEDIA '98: Proceedings of the sixth ACM international conference on Multimedia. New York, NY, USA: ACM, 1998. P. 235–240.
  28. *Wolfson H., Rigoutsos I.* Geometric hashing: An overview // IEEE Transactions on Computational Science and Engineering. 1997. Vol. 4, N. 4. P. 10–21.
  29. *Zhu Y., Shasha D.* Query by humming: a time series database approach // Proceedings of the ACM SIGMOD/PODS. 2003.
  30. *Zwislocki J. J.* Sensory Neuroscience: Four Laws of Psychophysics. Springer, 2009. 174 p.