УДК 004.93

АЛГОРИТМЫ РАСПОЗНАВАНИЯ МУЗЫКАЛЬНЫХ ОТРЫВКОВ, НОТ И АККОРДОВ

С.В. АНИСИМОВА, канд. физ.-мат. наук, доц. А.И. ГОЛОВАТЫЙ
(Белорусский государственный университет, Минск)

Рассмотрены существующие методы распознавания музыки, их применимость к распознаванию акапельного пения, а также способы их улучшения, одним из которых является транскрибирование звука в музыкальную нотацию. Предложены методы выделения нот и аккордов из звукового ряда.

Ключевые слова: *распознавание музыки, распознавание акапельного пения, выделение нот, выделение аккордов.*

Большинство современных приложений для распознавания музыки не умеют сопоставлять акапельное исполнение и оригинал записи. Изучение алгоритма распознавания одного из самых известных приложений, Shazam, дает возможность понять, почему так происходит.

Shazam Entertainment, Ltd была основана в 1999 г. Тогда же появилась идея создания сервиса для распознавания мелодий. На тот момент компания преследовала следующую цель: создать алгоритм, способный распознать мелодию, записанную при помощи простейшего микрофона телефона, отделяя ее от шума. Проблема заключалась в том, что на тот момент никто не делал ничего подобного. В первые годы развития компании, чтобы воспользоваться их приложением, необходимо было позвонить на короткий номер, повернуть телефон микрофоном к источнику звука и ожидать СМС с названием песни и исполнителем. В работе [5] в 2003 г. был описан алгоритм распознавания мелодий, который использует Shazam.

Принцип работы алгоритма «Shazam». Сначала записанный трек, который необходимо распознать проходит процедуру, в процессе которой извлекаются хеши. Полученный «отпечаток» мелодии сравнивается с базой данных, затем происходит уточнение хешей, процедура повторяется. Стоит отметить, что в базе данных аудиофайлы уже хранятся в хешированном виде. Выделение хешей включает в себя следующие этапы: на вход алгоритм получает частотно-мощностную временную зависимость; в ней он выделяет пики – наиболее «сильные» (англ. *robustness*) частоты среди шума и линейно аппроксимирует их по принципу суперпозиции; точка рассматривается как математическая с единственной парой координат. Точка является подозрительной на локальный экстремум, если она обладает большей энергией, чем точки в окрестности. Алгоритм составляет так называемую карту «созвездий» – график, на котором отмечены только локальные экстремумы (рисунок 1). В таком виде хранить записанную мелодию гораздо проще, это занимает меньше объема. Кроме того, пользователь записывает только небольшой отрывок аудио, а не всю песню, запись которой вдобавок зачастую изобилует шумами. Проблему шумов также необходимо было решить при создании алгоритма.

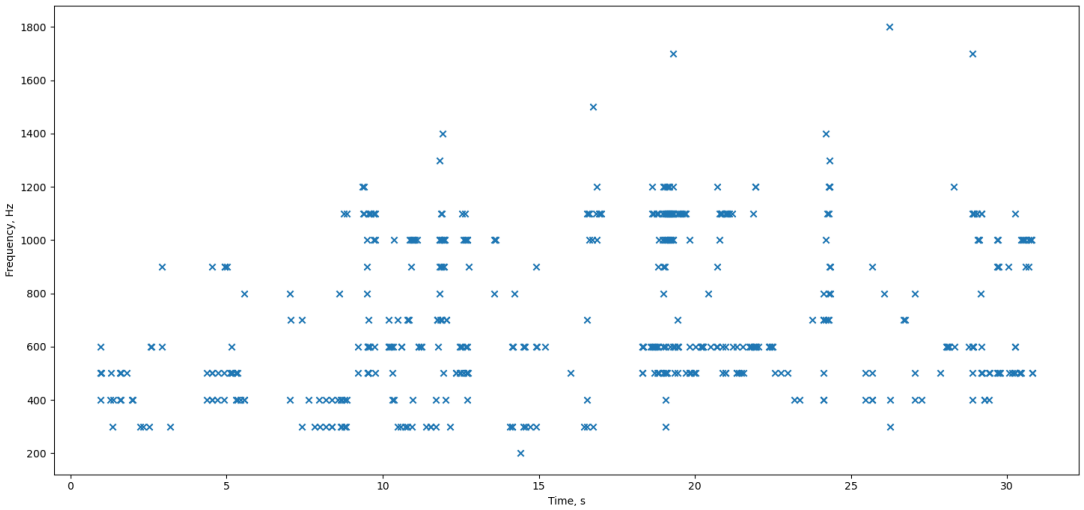


Рисунок 1. – Карта созвездий

Разработчики алгоритма пришли к достаточно простому решению. Они стали выделять точки – «якоря». Каждому якорю ставится в соответствие своя таргетная зона. Каждая точка внутри таргетной зоны

последовательно сопоставляется со своим якорем. Такую пару можно описать двумя частотами (якоря и точки внутри зоны) и временным интервалом между ними. Таким образом и происходит процесс хеширования. Данный алгоритм будем называть «быстрым комбинаторным хешированием», как в оригинальной статье «Fast combinatorial hashing» [5]. Также стоит упомянуть, что каждая строка в хеш-таблице может быть помещена в 32-битное беззнаковое число, что позволяет существенно сэкономить дисковое пространство.

Стоит отметить, что, конечно же, исходного кода самого Shazam в открытом доступе нет, и нет оснований достоверно утверждать, что в ходе выполнения практической части его удалось воспроизвести абсолютно точно. Однако Shazam здесь рассматривался не как конкретная разработка компании Apple Inc, а как «черный ящик», работающий по принципу, описанному выше.

Алгоритм Shazam не может распознать мелодию по отпечатку, получаемому из акапельной записи, так как количество локальных экстремальных точек в отпечатке оригинала будет сильно превосходить их количество в отпечатке напева. Исходя из этого, нами была выдвинута *гипотеза* о том, что, если применять алгоритм быстрого комбинаторного хеширования в диапазоне человеческого голоса, он должен работать корректно. На практике такой подход показал относительно хорошие результаты по сравнению с применением алгоритма без обрезки.

Алгоритм хеширования. В разработанном нами алгоритме хеширования каждая точка на «карте созвездий» является парой ключ-значение – время-частота экстремальных точек, взятых с определенным пороговым значением изменения энергетической характеристики звука (его громкости). Пороговым значением является выбранная разница уровня громкости сигнала между соседними точками в плоскости время – частота, которая считается достаточной для того, чтобы взять точку как экстремальную. С линейным увеличением порогового значения объем занимаемой одной записью памяти уменьшается экспоненциально. Сама «карта созвездий» представляет из себя матрицу булевых переменных. Поиск экстремумов в функции двух переменных был реализован с помощью свойства «diff» класса «DataFrame» библиотеки pandas (открытой python-библиотеки).

Поисковый алгоритм. Принцип работы поискового алгоритма: после получения хеш-таблиц оригиналов из базы данных и напетого фрагмента данные в каждой хеш-таблице покрываются объектами класса «Target», которые обладают следующими свойствами:

1) имеют размер 400 Гц на 5 секунд;

2) имеют свойство «target_property», являющееся ассоциативным массивом, или хеш-таблицей, в которой хранятся хеши, сформированные как $df : dt$, где df и dt – разницы частоты и времени между всеми точками объекта типа «Target» и точкой, обладающей наименьшей частотой и временем соответственно (т.е. точкой, имеющей наименьшее расстояние до точки отсчета в системе координат);

3) имеют перегрузку оператора «==». Для перегрузки оператора «==» необходимо выбрать два коэффициента: допустимое отклонение dt для того, чтобы считать два хеша одинаковыми – k_1; необходимое количество совпадающих хешей для того, чтобы считать два объекта типа «Target» одинаковыми – k_2. Степень схожести записей определяется отношением количества найденных равных объектов к их общему числу в капельной записи. Наибольшей схожестью в среднем являлась схожесть порядка 30–40%. Реализация приведена в [8; 9].

Однако результаты исследований данного алгоритма, которые описаны в [1] были все равно недостаточно хороши, даже с учетом «улучшения» за счет обрезки частотного диапазона до области человеческого голоса, поэтому необходимо было искать другие пути улучшения. Было замечено, что те записи, где тембры голосов акапельного и оригинального исполнителей сильно различались, практически не распознавались. Очевидно, эта особенность является одним из слабых мест этого алгоритма, поэтому оптимальнее было бы построить алгоритм, никак не зависящий от частотного диапазона певца. При этом очевидно, что если исполнитель не фальшивит, то он будет исполнять те же ноты, что и в оригинальной записи, но в другой октаве. Кроме того, исполнитель может петь быстрее, либо медленнее оригинала, однако отношение временных интервалов должно быть одинаковым для сохранения правильного ритмического рисунка композиции. Так, была выдвинута гипотеза о том, что если транскрибировать оригинал и акапельную запись в музыкальную нотацию и проводить сравнение по ним, то точность распознавания может быть повышена.

Распознавание нот. Задача распознавания отдельно стоящих нот сводится к двум подзадачам: непосредственному распознаванию ноты и распознаванию нот во временном интервале, т.е. отделении одной ноты от другой.

Алгоритм распознавания отдельно стоящей ноты достаточно прост. На первом этапе выделяется спектр звука с помощью преобразования Фурье (рисунок 2, *а*). Следующим шагом выделяются точки, подозрительные на локальные экстремумы (в данном случае интересуют только максимумы функции). Для этого реализована функция, в которой для каждой исследуемой точки вычисляется разница с двумя ее соседями. Если оба эти значения положительны, то в список записывается большее из двух значений. После этого выбирается первый экстремум, для которого значение в списке больше определенного, эмпирически подобранного значения толерантности. Далее берется полученное значение частоты и по нему определяется нота.

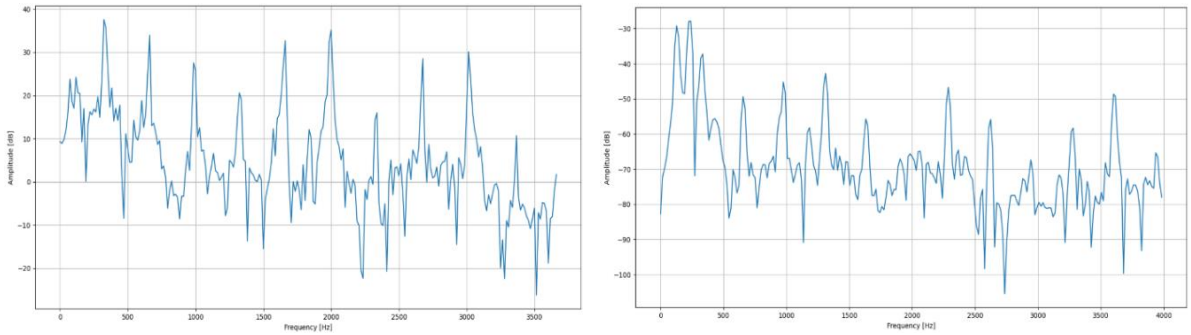
Чтобы не хранить весь список нот, а также чтобы в дальнейшем иметь возможность сличать одну и ту же мелодию напетую в разных диапазонах, т.е. не зависить от октавы в которой исполнена нота, она рассчитывается относительно 0-й октавы, то есть субконтроктавы. Поэтому в памяти хранится только небольшой ассоциативный массив на 7 нот: note_dict = {'E': 20.61, 'F': 21.82, 'G': 24.5, 'A': 27.5, 'B': 30.87, 'C': 32.7, 'D': 36.95}. Полученная на предыдущем этапе частота делится последовательно на каждое из этих чисел без остатка. Для того чтобы определить, какая именно нота попала на вход алгоритма, остается лишь определить какое из значений после деления является степенью двойки. Для этого достаточно вычислить результат побитовой операции «И» для полученного значения и для значения, на единицу меньшего, чем полученное. Результат применения оператора & будет равен 0 только для степени двойки и нуля.

Так же, вероятнее всего, для различных инструментов из-за особенностей их конструкций и звучания необходимо подбирать оконную функцию для выделения спектра. Например, для гитары такой функцией оказалась функция Ханна

$$w[n] = w_0\left(\frac{L}{N}\left(n-\frac{N}{2}\right)\right) = \frac{1}{2}\left[1-\cos\left(\frac{2\pi n}{N}\right)\right] = \sin^2\left(\frac{\pi n}{N}\right), \quad 0 \le n \le N.$$

На рисунке 2 видно, какой эффект привносит использование функции Ханна по сравнению со спектром, полученным без использования какой-либо оконной функции.

Этот механизм можно также применять для распознавания идущих подряд нескольких разных нот. Однако в этом случае уже просто выделить спектр сигнала на всем аудиофайле не получится. Для того чтобы понять, как разрешить данную проблему необходимо было рассмотреть амплитудно-временную зависимость, которую несложно получить, так как изначально аудиофайл считывается на входе в алгоритм функцией, которая сохраняет данные именно в таком формате. Нетрудно понять, что для отделения одной ноты от другой необходимо просто выделить сигнал среди шума. Для простоты реализации было принято, что сигнал можно назвать полезным, если он громче среднего по аудиофайлу на протяжении четверти секунды. Однако эти приближения нуждаются в коррекции и более тщательном подборе при анализе «живых данных».



а *б*

Рисунок 2. – Спектр ноты E4 без применения оконной функции (*а*) и с использованием функции Ханна для выделения (*б*)

Распознавание аккордов. При написании музыки используются не только одиночные ноты, но и аккорды, состоящие одновременно из нескольких нот, которые воспринимаются человеком как единый звук. Звук, сопоставляемый с определенным аккордом, является суммой гармоник. В реальности каждая последующая гармоника звучит тише предыдущей и хуже различима. Частота звука той или иной ноты определяется по ее основной гармонике.

Большинство современных алгоритмов транскрипции аккордов состоит из двух основных этапов. На первом этапе аудиозапись преобразуется в последовательность признаков векторов. На следующем этапе, используя шаблоны, каждый вектор признаков ставится в соответствие каждому аккорду из заданного пространства аккордов. Существует множество методик сопоставления с образцов на основе шаблонов, скрытых моделей Маркова или более сложных Байесовских сетях. Идея данного подхода заключается в том, чтобы заранее подготовить набор шаблонов, соответствующих набору аккордов в звуковом ряде. Интуитивно каждый шаблон в наборе – это прототип вектора цветности, который соответствует одному аккорду. На вход алгоритма подается звуковая дорожка, в которой, как и в алгоритме «Shazam», выделяются экстремальные точки. Однако для распознавания не берутся все точки, попавшие в карту созвездий, как в предыдущем алгоритме. К распознаванию берутся только те частоты, которые осциллируют вокруг одного значения некоторый промежуток времени. Каждой такой частоте ставится в соответствии с шаблоном нота.

Алгоритмы шаблонного распознавания основаны на сопоставлении конкретного аккорда из заданного аккордного пространства с входным вектором особенностей с помощью минимизации расстояния между векторами. Такую задачу можно решать с помощью алгоритмов кластеризации.

При распознавании нот и аккордов очень важна точность получаемых значений частот, так как один аккорд может состоять из нот с небольшой разницей в частотах. Поэтому для реализации нового алгоритма было принято решение написать выделение спектра, основываясь на быстром преобразовании Фурье, с возможностью самостоятельно задавать количество компонент разложения и получать при этом возможность исследовать влияние применения различных оконных функций.

В общем случае аккорд ведет себя схожим образом с нотами. Функция, описывающая наиболее «сильную» компоненту звука аккорда, также является периодической. Однако, рассматривая спектрограмму аккорда, был замечен интересный эффект: энергетическая мощность гармоник изменяется по какому-то периодическому закону, как при генерации аккорда с помощью программных средств пакета Wolfram Mathematica 11.3, так и при воспроизведении аккорда на гитаре. Это происходит из-за того, что при сложении нескольких близких по частоте волн наблюдается эффект биения.

Биения возникают при сложении нескольких волн одинакового направления, мало отличающихся по частоте [3]:

$$x_1 = a\cos\omega t,$$

$$x_2 = a\cos(\omega + d\omega)t;$$

$$x = x_1 + x_2 = \left(2a\cos\frac{d\omega}{2}t\right)\cos\omega t.$$

Результирующая частота звука рассчитывается как среднее частот нот. На рисунках 3, 4 можно увидеть результирующую функцию суммы периодических функций, составляющих основные гармоники нот аккорда С.

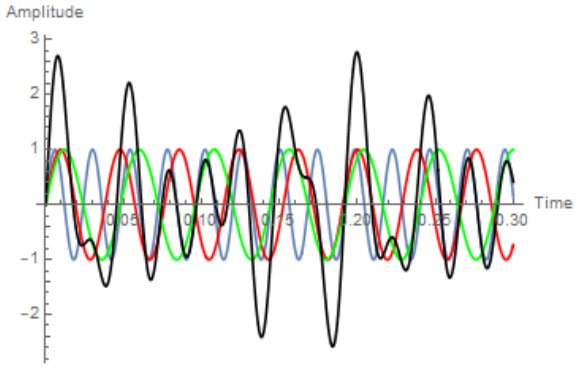


Рисунок 3. – График периодических функций основных гармоник нот аккорда С и их суммы

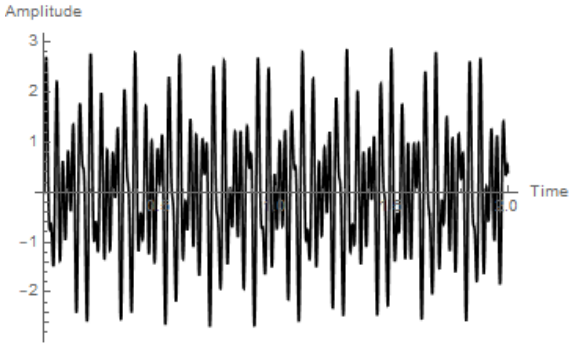


Рисунок 4. – График суммы основных гармоник нот аккорда С

Из этого следует, что при реализации алгоритма не удастся выделять три ноты раздельно и по ним опознавать аккорд. Решением данной проблемы является создание базы признаков аккордов или распознавание аккорда по его результирующей частоте.

На практике оказалось, что оба эти метода применимы. Действительно, иногда в аккорде удается выделить результирующую частоту, однако данный метод нестабилен и подвержен влияниям шумов, а также на его результативность сильно влияет выбранное количество компонент разложения Фурье.

Другим эффективным методом является создание шаблонов характеристик интересующих аккордов. Способов выделения векторов особенностей существует очень много. В работе [6] доказывалась эффективность выделения векторов цветности. Кроме того, можно предложить еще два варианта создания шаблона: создание «звездной карты» по методу «быстрого комбинаторного хеширования», которое было описано в работе [5], и создание вектора особенностей аккордов по методу, рассмотренному во время анализа распознавания отдельных нот с дополнительными оговорками. Если для распознавания отдельной ноты достаточно выделить одну первую основную гармонику, то есть глобальный экстремум, то для распознавания аккордов необходимо было выделить 6–10 локальных экстремумов спектра.

Однако эти точки также не могут быть гарантированно такими же, как и в векторе свойств. Инструмент, на котором исполнен аккорд, может усиливать те или иные частоты в зависимости от свойств его резонаторов, а также свой вклад может вносить записывающее оборудование. Кроме того, сам исполнитель может сыграть аккорд недостаточно чисто. Поэтому из имеющихся эталонных векторов выделяются наиболее схожие компоненты и уже относительно них вычисляется расстояние до изучаемого вектора. Необходимо отметить, что данный алгоритм является устойчивым к смещению диапазона исполнения, так как опирается не на конкретные частные значения частот, а на Евклидову метрику.

Заключение. В заключение можно сказать, что при доработке методика распознавания нот и аккордов может быть применена для распознавания акапельного пения. Кроме того, исходя из исследований алгоритма «Shazam» можно сделать вывод, что обрезка частотного диапазона оригинала до диапазона человеческого голоса (до 4000 Гц) может помочь в распознавании музыки. Немаловажным шагом в разработке алгоритма распознавания акапельного пения стал анализ составляющих музыкального ряда, а также методов работы с ними. На данный момент разработаны алгоритмы выделения нот и аккордов, которые необходимы для создания алгоритма распознавания мелодий на основе музыкальной нотации. Для успешной разработки такого алгоритма также необходимо решение для распознавания аккордов среди нот и наоборот.

ЛИТЕРАТУРА

1. Анисимова, С.В. Алгоритм распознавания музыкальных композиций по вокальному исполнению / С.В. Анисимова, А.И. Головатый, И.А. Адуцкевич // Информационные радиосистемы и радиотехнологии : материалы респ. науч.-техн. конф., Минск, 28–29 окт. 2020 г. / Белорус. гос. ун-т информатики и радиоэлектроники ; редкол.: В.А. Богуш [и др.]. – Минск, 2020. – С. 185–186.
2. Мефферт, Б. Инструменты обработки сигналов – основы, примеры применения и задачи / Б. Мефферт, О. Хохмут. – Берлин : HUB, 2019. – 320 с.
3. Савельев, И.В. Курс общей физики : в 3 т. / И.В. Савельев. – М. : Наука, Главная редакция физико-математической литературы, 1970. – Т. 2 : Механика, колебания и волны, молекулярная физика. – 517 с.
4. Dan Li. Design of far field beamforming problem by window function / Dan Li, Xiaofen Tan, Zhiguo Feng // IOP Conference Series : Materials Science and Engineering. – 2019. – Vol. 677, iss. 4.
5. Li-Chun Wang. An Industrial-Strength Audio Search Algorithm [Electronic resource] / Li-Chun Wang // ISMIR 2003 : Proceedings of 4th International Conference on Music Information Retrieval, Baltimore, Maryland, USA, October 27–30, 2003. – Mode of access: https://www.researchgate.net/publication/220723446_An_Industrial_Strength_Audio_Search_Algorithm. – Date of access: 17.05.2020.
6. Analyzing Chroma Feature Types for Automated Chord Recognition / Nanzhu Jiang [et al.] // Proceedings of the AES 42nd International Conference: Semantic Audio. – Ilmenau, Germany : AES, 2011. – P. 285–294.
7. Веб-сервис для хостинга IT проектов [Электронный ресурс] – Режим доступа https://github.com/Allen-Downey/ThinkDSP. – Дата доступа: 14.06.2013.
8. Веб-сервис для хостинга IT проектов [Электронный ресурс] – Режим доступа https://github.com/sofiaandrat/soundAnalyze_termPaper2020. – Дата доступа: 17.05.2020.
9. Веб-сервис для хостинга IT проектов [Электронный ресурс] – Режим доступа https://github.com/sofiaandrat/Finding_termPaper2020. – Дата доступа: 17.05.2020.

Поступила 30.04.2021

ALGORITHMS TO RECOGNIZE MUSICIAL EXCERPTS, NOTES AND CHORDS

S. ANISIMOVA, A. HALAVATYI

The paper discusses existing methods of music recognition, its adaptability to recognition of acapella singing and methods of its improvement. One of the improvement methods is sound transcription to musical notation. Methods of note and accord extraction from sound series were proposed.

Keywords: *Music recognition, acapella singing recognition, extraction of notes, extraction of chords*.