Enumeration of Chord Sequences

Nick Collins
University of Sussex
N.Collins@sussex.ac.uk

ABSTRACT

The enumeration of musical objects has received heightened attention in the last twenty five years, and whilst such phenomena as tone rows, polyphonic mosaics, and scales have been explored, there has not been prior investigation of the enumeration of chord sequences. In part, analysts may have disregarded the situation as having a trivial solution, namely the number of chord types at each step raised to the power of the number of steps. However, there are more subtle and interesting situations where there are constraints, such as rotational and transpositional equivalence of sequences. Enumeration of such chord sequences is explored through the application of Burnside's lemma for counting equivalence classes under a group action, and computer generation of lists of representative chord sequences outlined. Potential extensions to 'McCartney's Chord Sequence Problem' for the enumeration of cyclic (looping) chord sequences are further discussed.

1. INTRODUCTION

From D. L . Reiner's enumerations of set and row classes [1] to Harald Fripertinger's investigations of mosaics and motifs [2], the application of combinatorial methods in music theory has advanced considerably. An excellent review article by Julian Hook [3] presents an introduction to the use of such theoretical apparatus from combinatorics as Burnside's lemma, [1] and Pólya's enumeration theorem; the reader might also consult David Benson's textbook on music and mathematics [4], and from a more intensely mathematical perspective, a concise survey by the combinatoricist R.C.Read [5].

This paper tackles the counting and generation of all chord sequences meeting certain musical criteria. The motivation for this work was a question raised, and partially answered, by James McCartney in a post on the facebook social networking site. He was curious as to the number of possible 3 chord sequences, taking into account rotational and transpositional equivalence, where the only chord types are major and minor, and repetitions are excluded. He wrote a program to enumerate the options by brute force [6]. The Facebook post became a site of discussion, with respondents quickly noting the applications in automatic pop song generation. Chandrasekhar Ramakrishnan provided an argument to enumerate the 3 chord case, based on looking at successive intervals. Whilst this elucidates the situation for a small number of chords, it does not generalise so well. This author posted a solution to the more general case of a sequence of L chords from C types of chords within an (equal tempered) pitch class universe of N pitches. This paper expands upon this more general solution, and the issues that arose in consideration of this problem.

[1] With appropriate acknowledgement of the controversy over this attribution, nodding to Cauchy and Frobenius.

Additional musical factors can complicate the picture, from voice leading through to the details of timing in chord presentation (whether inter chord time interval or arpeggiation delays within an individual chord). We shall assume herein that chords are presented without respect to duration, that is, as if the chord sequences are pure entities of interest in themselves (which is a good approximation to much typical usage in composition, especially where harmonic rhythm is not especially varied, for example pop song construction). In many cases below, the spread of chord notes over multiple octaves can be abstracted away either by the notion of a dictionary of chord types (which is taken to include all possible voice configurations) or through pitch class set reduction.

There is a basic and effective solution to enumerating chord sequences, which is detailed here as a starting point. This is to consider that order of presentation in time is fundamental to practical revelation in music, so rotational equivalence does not need to be invoked. Further, though naive transpositional equivalence between chord sequences may be necessary (absolute pitch is relatively rare), each sequence is distinct up to no more than N possible transpositions within a given pitch space. [2] For a sequence of length L, with C chord types available (each having N non-equivalent transpositions) there are then $\frac{(C*N)^L}{N}$ possible chord sequences. The rest of this paper looks at various ways in which this initial picture might be complicated. The investigation is musically productive, revealing various interesting structures (such as auto-mapping sequences) that may be inspiring to composers. A further counter argument to to those who might restrict themselves only to studying fixed sequences might highlight principles of eternal cyclic return in many world musics, or minimalist works. The categorisation under constraints on chord repetition discovers some interesting chord sequence rep-

[2] We ignore for now the existence of sequences of chords entirely made from self-transposing chords such as diminished seventh, augmented or whole tone types admitting less than the full N transpositions (what Messiaen called 'modes of limited transposition').

resentatives; rotations can always be trivially restored from the discovered equivalence classes.

Previous work on chord sequences extends from encoding and probabilistic modeling for music information retrieval [7, 8], through applications in algorithmic composition [9, 10] to more mathematical music theoretic approaches [11]. In the latter paper in particular, Clifton Callendar and colleagues [11] allude to chord sequence enumeration within the context of voice leading, based on their geometric construction for n-note chords in an n-dimensional space. However, their 'individual equivalent' spaces in particular allow the association of any chord with any other, and are too loose for our purposes, whereas for 'uniform' transformations (the same operation applied to all sequence members), they do not consider rotation of sequences themselves within their OPTIC set of musical transformations (since these transformations must apply to an individual object of a sequence, but not act as permutations on a sequence).

On a mathematical note, the problems considered in this paper overlap to some extent with the formulations for k-sequences in combinatorial algorithms [12, p.226]. However, the rotation of chord sequences, and the existence of more than one chord type, takes us away from considering a k-sequence of integers. The set X is that of chord sequences themselves where each position in the sequence has an associated tuple of transposition and chord type. Although we shall see a useful encoding in section 3 that drops back to a sequence of integers for the purposes of establishing an order relation, the tuple is still implicit. Similarly, treatments of rows, even with rotational equivalence, or 'Stockhausen's Problem' of enumerating possible performances for an open form work [5], do not match the chord sequence problem as elucidated by James McCartney, and we now proceed to tackle this problem directly.

2. EQUIVALENCE CLASSES OF CHORD SEQUENCES UNDER TRANSPOSITION AND ROTATION

Consider a set X of chord sequences of length L, within an equal tempered pitch class space of size N (the N-scale in Fripertinger's notation [2]). The group of transpositions on the space is Z_N, whose entries are also the integers 0 to N-1 just like the pitch classes; in practice, this should not cause confusion. Chords are taken from a fund of M types, where the set M takes in all possible operational modifications of chords, for example, the set of major and minor chords at all transpositions (but without inversional equivalence). In other cases, M might be much larger: for instance, in the representation used by Absolu, Li and Ogihara [8] where base chords can have various modifiers on the 3rd, 5th, 6th/7th, 9th, 11th, 13th, and an additional bass tone, the authors observe $M = 331776$ possibilities.

We want to enumerate all equivalence classes of chord sequences under a group action, from a group generated by primitive operations:

- rotation R: cyclic rotation of sequence by one step to the right, wrapping around the last entry to the first
- transposition T: transposing all elements of the sequence by one step. T is the smallest transposition step in Z_N

Powers of the generators lead to rotation by an arbitrary number of steps (up to R^L as the identity) and transposition by an arbitrary amount up to T^N as the identity. The complete group G is co-generated by R and T (which are commutative as operators on sequences, in that their action is independent [13]) and is isomorphic to $Z_L * Z_N$, a group of order LN.

For now we exclude retrograde operations on chord sequences, which would otherwise lead to a product of a dihedral group and cyclic group. We shall, however, consider the absence or inclusion of a number of additional constraints on valid chord sequences, such as a restriction that no chord is directly repeated. [3] .

In this context, a natural tool to employ is Burnside's lemma [2] [4, p.346], which states that we can count all distinct orbits (equivalence classes) by looking at the average number of fixed points across all group members:

$$\frac{1}{|G|}\sum_{g\in G} fix(g,X) \tag{1}$$

where $fix(g,X)$ is the number of elements of X left fixed under application of the action of g.

Burnside's lemma tells us to look for fixed points of the group of (rotation, transposition) operators acting on the set X of possible chord sequences. We must think about the logic of possible sequences which map to themselves under particular rotations and transpositions.

2.1 A motivating example

To explicate the general solution to be outlined in section 2.2, consider some specific cases first, based around three chord sequences. The sequence

C_1, C_2, C_3

maps to itself under rotation R iff $C_1 = C_2 = C_3$. If we have a further condition that repetition is not allowed in sequences, then this case has already been excluded. Otherwise, there are M possible sequences of this type (M choices for the particular repeated chord). To map to itself under a single step transposition T, $T(C_i) = C_i$ for all i. If we assumed that the chords were already distinct, this is not possible. Indeed, the only candidate chord otherwise self-equivalent under simple T is the aggregate chord of the complete chromatic pitch space (the complement of the pathological empty chord), which is unlikely to be included in more practical sequences, for example, of three note major and minor chords in Z_{12}! For the moment, we will side step such cases of self-transposing chords as diminished sevenths or augmented.

The general operator case R^aT^b for some a, b is more interesting, and key to the solution below. Here, chords in the sequence are identified within particular subsequences,

[3] In terms of varying harmonic rhythm, chords might very well be allowed to repeat, for instance the twelve bar sequence of major chords C C C C F F C C G F C C; if repetition is excluded, this sequence drops down to C F C G F

determined by the relationship of a to L. For example, for our three chord case, $L = 3$ is prime, so any a must lead to an association across the complete sequence. However, for composite numbers L, there can be more sophisticated subsequence behaviour. If L was 6 and a 2, then two subsequences indexed 0,2,4 and 1,3,5 respectively would be associated. The question is then whether the chain of transpositions based on b can 'fit' within the length of the derived subsequences. For example, for sequences of six major chords in Z_{12}, $a = 2, b = 4$ would be permissible, leading to such fixed elements of X under R^2T^4 as:

C, E♭, E, G, A♭, B

and many more, based on free choice of starting chords up to additional constraints on repetition of successive chords (see Figure 1). It is seen that the orbits of chords under particular transpositions interact with the size of orbits of a within Z_L.

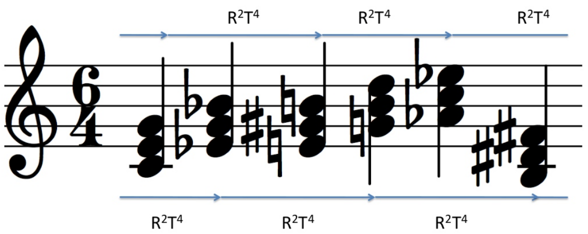

Figure 1. **Six chord sequence** Demonstrating two subsequences with R^2T^4 equivalence.

2.2 General solution

We now state the general solution. The number of possible looping chord sequences is:

$$\frac{\sum_{i,j \text{ s.t. } condition(i,j)} summand(M, L, i, j)}{NL}$$

$$i = 0, \ldots, N-1 \quad j = 0, \ldots, L-1 \tag{2}$$

where $condition(i, j)$ is that

$$i * order(j) \equiv 0 \bmod N \tag{3}$$

($order(j)$ being the order of j in the cyclic group of order L, Z_L). i and j try out all possible transpositions and rotations; only a subset of these have any possibility of setting up actual fixed points under the group operations. The case $i = 0, j = 0$ of course is the case for $fix(I, X)$ where I is the identity operation (no rotation, no transposition).

The condition then states that there is 'room' for a sequence of transpositions to follow the orbit spiral of element j within Z_L and end up exactly back where they started, as motivated in section 2.1.

The summand depends on what variant of strictness we're using for consecutive chords, and expresses the amount of freedom on substituting chord choices. Let the number of orbits of j within Z_L be denoted $orbits(j) = L/order(j)$

For a free choice:

$$summand(M, L, i, j) = M^{orbits(j)} \tag{4}$$

i.e., M to the power of the number of orbits of j in cyclic group Z_L.

For the cases disallowing consecutive repetition of the same chord (of specific type and transposition), or between any chord type sharing the same transposition level, things get more complicated, but a recursive solution still exists. Define Q as either 1, for disallowing consecutive chords of the same type and transposition, or as the number of distinct chord types independent of transposition (for example, when $M = Q*N$). Then to count the number of fixed elements given 'space' of $n = orbits(j)$ orbits within Z_L, consider the recursions:

$$\begin{aligned}
a_{M,Q}(2) &= M*(M-2Q) \\
a_{M,Q}(3) &= M*Q*(M-Q) + \\
&\quad (M*(M-2Q)*(M-2Q)) \\
a_{M,Q}(n) &= (M-2Q)*a_{M,Q}(n-1) + \\
&\quad Q*(M-Q)*a_{M,Q}(n-2) \quad n \geq 3 \\
b_{M,Q}(2) &= M*(M-Q) \\
b_{M,Q}(3) &= M*(M-Q)*(M-2Q) \\
b_{M,Q}(n) &= (M-2Q)*b_{M,Q}(n-1) + \\
&\quad Q*(M-Q)*b_{M,Q}(n-2) \quad n \geq 3
\end{aligned} \tag{5}$$

where the final summand follows as:

$$\begin{aligned}
summand(M, L, i, j) &= 0 \\
&\qquad orbits(j) = 1, i = 0 \\
summand(M, L, i, j) &= M \\
&\qquad orbits(j) = 1, i > 0 \\
summand(M, L, i, j) &= a_{M,Q}(orbits(j)) \\
&\qquad orbits(j) > 1, i > 0 \\
summand(M, L, i, j) &= b_{M,Q}(orbits(j)) \\
&\qquad orbits(j) > 1, i = 0
\end{aligned} \tag{6}$$

Figure 2 provides an illustration to accompany the justification for this formula. The motivation is that we have $n = orbits(j)$ slots to assign to, into which we can ostensibly substitute any of M chords. However, we must restrict choices based on neighbouring chords to avoid certain forms of repetition. Rather than proceed assigning chords from the start of the slots, we proceed in reverse and seek a recursion that expresses the solution for n in terms of the solution for smaller numbers of slots. Note that the recursive formulas for $a_{M,Q}(n)$ and $b_{M,Q}(n)$ are the same, but with different initial conditions; we shall resolve this below, though the first part of the argument concerning the recursion itself works in both cases.

The slots are labelled in the diagram as $C_1, C_2, \ldots, C_{n-1}, C_n$ and an additional element of C_{n+1} representing the next element in the subsequence orbit from C_1 (i.e., T^iC_1), or wraparound back to C_1. In practice, which is the case does not matter to the argument. It is sufficient to consider that the number of choices for C_n depend on the occupancy of the slots next door. We have to deal with two cases.

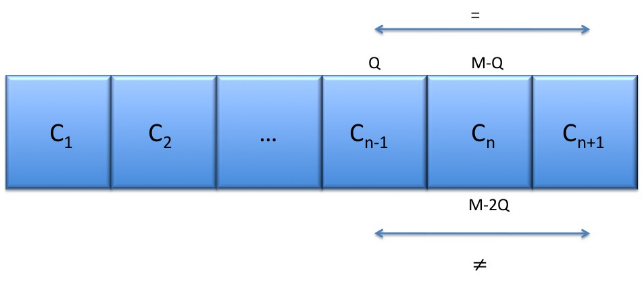

Figure 2. **Chord choice recursion construction** Considering chord choice across orbits

In the first, $C_{n-1} = C_{n+1}$. Here, there are Q choices for C_{n-1}, in that it must match transposition and chord type or just the transposition of C_{n-1} (this is the definition of Q). But since the two adjacent elements are from the same Q options, there are $M - Q$ choices left for C_n, and having selected both C_{n-1} and C_n, we can proceed to consider the slot at $n-2$, providing the $Q*(M-Q)*a_{M,Q}(n-2)$ term in the recursion.

In the second case, $C_{n-1} \neq C_{n+1}$. Now for C_n there are $M - 2Q$ options. We have not fixed slot $n-1$, and we already know it must not be equal to slot n, so the recursion is now to consider this element $n-1$, and the term in the recursion here is $(M-2Q)*a_{M,Q}(n-1)$. This completes the proof of the recursion itself.

We do have to take account of some different initial conditions; a corresponds to the situation where $C_1 \neq C_{n+1}$, and b to $C_1 = C_{n+1}$. The reader should be able to reproduce the formulas by consideration of the base cases for a small number of slots. For example, for the most complicated, $a_{M,Q}(3), C_1 \neq C_4$ (C_4 is in the same orbit as C_1 but is at a new transposition level, since $i > 0$). The two terms in the sum come from two cases; first, if $C_2 = C_4$ there are M choices for the first slot, Q for the second (since it must equal in transposition at least that of the wrap element at the fourth), and since either side of C_3 is at one transposition level, $M - Q$ choices for the third. The other term in the sum follows analogously as M for the first, $M-2Q$ for the second (not equal to first or fourth) and then the third is not equal to the second or third, allowing $M - 2Q$ options again. [4]

[4] The reader should hopefully be convinced already. But the author arrived at this result via a more complicated pair of joint recursions for situations where the $n+1^{th}$ element was equal to C_0 or different; it eventually turned out that the simpler recursion presented here could be derived, though the distinction lives in the initial conditions.

$$\begin{aligned} a_{M,Q}(n) &= Q*b_{M,Q}(n-1) + (M-2Q)*a_{M,Q}(n-1) \\ &\qquad n \geq 3 \\ b_{M,Q}(n) &= Q*(M-Q)*b_{M,Q}(n-2) + \\ &\quad (M-2Q)*(M-Q)*a_{M,Q}(n-2) \\ &\qquad n \geq 4 \\ a_{M,Q}(2) &= M*(M-2Q) \\ b_{M,Q}(2) &= M*(M-Q) \\ b_{M,Q}(3) &= M*(M-Q)*(M-2Q) \end{aligned} \quad (7)$$

We assumed in the above derivation that there were no self-transposing chords. Self-transposing chords have more than a single orbit when acted upon by Z_N, so they repeat for some transposition less than N. For example, a diminished seventh chord within 12TET repeats within a three semitone transposition (there are three orbits), an augmented chord within four. This leads to non-empty fixed sets for transpositions without rotation, as long as only appropriate self-transposing chords are selected in the sequence; the presence of any non-self-transposing chord breaks the spell. It also allows for additional fixed sequences under rotation and transposition. Though M may be correspondingly reduced to start with, and some of the argumentation above remains valid (only taking up to $2Q$ off M in the recursion may seem to cover us up to self transposing chords with at least two distinct forms), the situation is complicated especially when including a number of different self-transposing chords. No analytical solution has yet been perfected here.

2.3 Worked examples

For the problem originally proposed by James McCartney, $L = 3$, $N = 12$, and $M = 24$ (major and minor chords over $N = 12$ transpositions). The overall group size is $NL = 12*3 = 36$

To find orbits within which successive transpositions fit, we need all pairs (i, j) such that $i * order(j) \equiv 0 \bmod 12$ where $i = 0, \ldots, 11 \quad j = 0, 1, 2$. The orders of j within Z_3 are 1,3 and 3 respectively, so the corresponding numbers of orbits are 3, 1 and 1. Solutions for i solving the modulo congruence are $i = 0$ for $j = 0$, $i = 0, 4, 8$ for $j = 1$, $i = 0, 4, 8$ for $j = 2$. These values of i and j can be related to the generators R and T for the group acting on the set of chord sequences as the identity I, R, RT^4, RT^8, R^2, R^2T^4 and R^2T^8.

From this point, the different constraints on consecutive chords come into play. For free choice, we get 24^3 sequences fixed by the identity, and a contribution of $6*24$ for the other six orbits of size 1. The whole summation is divided by the overall group size of 36, to get 388 distinct chord sequences. For no consecutive chords at the same transposition level, we get $24*23*22 + 4*24$ all divided by 36 is 340 (the two cases where the number of orbits is 1 and $i = 0$ end up contributing zero to the summation, since they have 'no space' to put chords all at the same transposition level; there is only one orbit). For no consecutive chords of any type at the same transposition, we have $24*22*20 + 4*24$ all divided by 36 is 296 chord sequences. Note that in this case, simpler reasoning than above can deal with the counts, since the length is only three; lengths greater than three require the more complicated formulas above.

To now work an example for an alternative musical space, let us consider 6 chord sequences in the diatonic space of 7 notes, as per Clough's discussion of diatonic sets [14]. The three non-congruent chord types will be the triad indexed [0,2,4], the 'suspended second' indexed [0,1,4] and the diatonic cluster [0,1,2]. We want to find the number of distinct chord sequences where there is no repetition of transposition level regardless of type, $L = 6$, $N = 7$, $M = 21$, $Q = 3$. The orders of j=0,1,2,3,4,5 within Z_6 are 1,6,3,2,3,6 respectively, so the corresponding numbers of orbits are 6,1,2,3,2,1. Solutions for i solving the modulo congruence are only $i = 0$ for each j (7 is prime). The overall summation works out then depending on the size of orbits and the formulas above as $(2 * b_{21,3}(6)) + b_{21,3}(3) + (2 * b_{21,3}(2))$ since the orbits of size 1 at transposition 0 contribute no valid sequences. Using the recursions to solve this, there are 810072 distinct chord sequences in this scenario.

An example which is very hard to enumerate exhaustively by computational search is $L = 50$, $N = 19$, $M = 57$ (three chord types, $C = 3$, $Q = 3$). Brute force enumeration would hunt through 57^{50} cases, more than the number of atoms in the observable universe. We can still count the number of musical objects even if running a program to generate them exhaustively would take too long; we reduce the hunt to looking at all pairs (i, j) where $i = 0, \ldots, 18 \quad j = 0, \ldots, 49$ and working out the appropriate summand, if valid. Calculating the summands takes us well out of the range of 64-bit long integers, so we need to use a special maths library. Python was used to run this particular calculation, as well as check others above, since it natively supports integers of arbitrary size. The answer isn't small:
438494737719887413398130512078099761413309
435496154668961729757520348645372634586590
distinct chord sequence equivalence classes, around 1/746th of the $57^{50}/19$ raw possibilities. [5]

3. COMPUTATIONAL GENERATION OF CHORD SEQUENCE EQUIVALENCE CLASSES

Whilst we have so far exhibited a general formula to count chord sequences under rotational and transpositional equivalence, we have not exhibited lists of representatives for each orbit. There is a natural encoding for sequences, which can be used to select a representative for an orbit as the sequence with minimal code. Let the chord at transposition level k ($k \in 0, \ldots, N-1$) and chord type q ($q \in 0, \ldots, Q-1$ for Q distinct chord types) be given a single number $k * Q + q \in 0, \ldots, M-1$ rather than an ordered pair. The sequence encoding is to take the ordered sequence of these L numbers each from 0 to $M-1$, and convert it to a Big Endian (most significant digit first) number in base M. That is, the first element of the sequence is the most significant digit:

$$\mathit{code}([x_1, x_2, \ldots, x_L]) = \sum_{i=1}^{L} x_i M^{L-i} \tag{8}$$

Brute force enumeration can then be programmed by trying all possible assignments of numbers to sequences (M^L possibilities), generating for each the orbit, and keeping those sequences which have minimal code within their orbit. Each code is unique, so the representatives will only turn up once.

In practice, we only need to consider sequences which have their first chord at transposition level 0, since this is a necessary condition for the minimal code representative (we can always transpose any given chord sequence to one which has a first chord at transposition 0). Note that as a consequence of our encoding, representatives containing earlier chord types (by the arbitrary numbering) will tend to be selected more often; the last chord type will only appear in the first slot for a representative if all chords in the sequence are of that type (for otherwise, rotate till another chord type is in the first slot, transpose to 0, and note this gives a lower code).

The restriction of the first element to one of Q rather than M numbers also enables a speed up in calculation by early return from testing members of an orbit. For each possible rotation of j chords, we check the $(L-j)$th element in the sequence. There is only one transposition i which will take it to transposition 0. Compare the chord type to that of the code to test at the first chord position. The candidate's first element must have a lower or equal code to survive as a candidate. Continue checking all other comparable entries (the transposition now being known), and then repeat the procedure for all possible rotations from 1 to $L-1$. At most $(L-1) * L$ numbers will be compared, as opposed to $L * L * M$ for the complete set of transpositions and rotations.

A better algorithm for generation has not yet been discovered. Although the author spent some time trying to apply R. C. Read's method of orderly generation [15], no encoding has been found to meet the more stringent criteria for an orderly algorithm; essentially, the tuples of transposition and chord type and the 'natural' single number encoding breaks Read's conditions. [6]

The number of orbits found empirically via the computer program matches the test cases described in section 2.3 (but for the 50 chord sequence example designed to be too difficult for brute enumeration). It is conceivable to adapt the program for situations with self-transposing chords; though additional transpositions must be checked, we can reduce the size of M to begin with to hunt through less possibilities. In general, this would require for each chord type an associated size of orbit under transposition. Although the coding may still utilise single numbers, with posthoc reductions when interpreting codes into chord sequences, it would probably prove more beneficial to deal with arrays of integers directly here.

4. CONCLUSIONS

We have investigated the enumeration of chord sequences under equivalences of transposition and sequence rotation, finding a solution for the counting of such sequences when

[5] If we allow $C = Q = 10000$, imagining a more involved set of chord types, the result has 260 decimal digits, and is around 10^{177} times larger!

[6] With respect to Read's variable q in the general problem [15, p. 112], this seems so both for working recursively from the sum to q over elements of the sequence and from q taken as the length of sequences, with augmentation by extension of sequence length.

there are no self-transposing chords, and considered the practical generation of lists of representatives. Aside from mathematical interest, and conceptual music theory, there is a real application in the generation of such sequences for algorithmic music.

Lest we lose sight of musical realities in mathematics, consider a situation which is perhaps closer to traditional musical practice. You are at a particular chord in an ongoing sequence, and must choose the next chord. Depending on stylistic convention, there is a limited set of available choices in the context. For instance, for Western common practice music, the context might be a dependency on the current key and recent chords, and we might consider a probability distribution over next chord options. Given an A minor chord in the key of C, there will be various standard following chords remaining within the key of C, as well as options setting up local or longer term modulations to the relative minor, G major, F major and more. The history of Western music sees a gradual increase in the freedom of modulation. This suggests an examination of more local conditions on selection of chords, which leads to an enumeration problem which is perhaps best explored by computer. However, analogously to the current solution for counting, a recursive solution might be pursued based on decisions with a remaining sequence length of L leading to decisions of length $L - 1$.

Other extensions to the chord sequence problem considered in this paper could involve:

- Further consideration of the issues of self-transposing chords.
- Further equivalence of sequences under the retrograde operation (expanding out to $Z_L * D_{2N}$, the product of a cyclic group and a dihedral group).
- Modifications for non-equal-tempered pitch class spaces where transpositional equivalence is less straight forward, or only operates for certain subsets. The transposition operation under such conditions may not lead to a valid group and group action.
- The combination of durations with chords (see for instance [8])

In terms of mathematical music theory, the consideration of McCartney's Chord Sequence Problem has been productive, and much potential remains for future work. Python code for counting equivalence classes, and C++ code for generation of lists of sequence representatives (for relatively small L) is made available to accompany this paper, and can be obtained from the author's web pages `http://www.sussex.ac.uk/Users/nc81/code/chordenumeration.zip`.

Acknowledgments

A huge thankyou to James McCartney for raising the challenging topic of chord sequence enumeration in the first place.

5. REFERENCES

[1] D. Reiner, "Enumeration in music theory," *Amer. Math. Monthly*, vol. 92, no. 1, pp. 51–54, 1985.

[2] H. Fripertinger, "Enumeration and construction in music theory," in *Diderot Forum on Mathematics and Music*, Vienna, 1999, pp. 170–203.

[3] J. Hook, "Why are there twenty-nine tetrachords? a tutorial on combinatorics and enumeration in music theory," *Music Theory Online*, vol. 13, no. 4, 2007.

[4] D. J. Benson, *Music: A Mathematical Offering*. Cambridge: Cambridge University Press, 2004. [Online]. Available: http://www.maths.abdn.ac.uk/~bensondj/html/maths-music.html

[5] R. C. Read, "Combinatorial problems in the theory of music," *Discrete Mathematics*, vol. 167/168, pp. 543–551, 1997.

[6] J. McCartney, "Number of unique 3 chord cyclic progressions," 2010. Facebook (login required). [Online]. Available: https://www.facebook.com/note.php?note_id=141375232545583

[7] J.-F. Paiement, D. Eck, and S. Bengio, "A probabilistic model for chord progressions," in *International Symposium on Music Information Retrieval*, 2005, pp. 312–319.

[8] B. Absolu, T. Li, and M. Ogihara, "Analysis of chord progression data," in *Advances in Music Information Retrieval*, Z. W. Raś and A. A. Wieczorkowska, Eds. Berlin: Springer-Verlag, 2010, pp. 165–184.

[9] C. Ames, "The markov process as a compositional model: A survey and tutorial," *Leonardo*, vol. 22, no. 2, pp. 175–187, 1989.

[10] G. Nierhaus, *Algorithmic Composition: Paradigms of Automated Music Generation*. New York, NY: Springer-Verlag/Wien, 2009.

[11] C. Callender, I. Quinn, and D. Tymoczko, "Generalized voice-leading spaces," *Science*, vol. 320, no. 5874, pp. 346–348, 2008.

[12] D. L. Kreher and D. R. Stinson, *Combinatorial Algorithms: Generation, Enumeration and Search*. Boca Raton, FL: CRC Press, 1999.

[13] R. D. Morris, *Composition with Pitch Classes*. New Haven, CT: Yale University Press, 1987.

[14] J. Clough, "Aspects of diatonic sets," *Journal of Music Theory*, vol. 23, no. 1, pp. 45–61, 1979.

[15] R. C. Read, "Every one a winner or how to avoid isomorphism search when cataloguing combinatorial configurations," *Annals of Discrete Mathematics*, vol. 2, pp. 107–20, 1978.