ABSTRACT
Harmonization is an important component of any comprehensive music generation system. Harmony provides context for melody and indicates the tonal framework and progression of a musical passage. One interesting element of music is that harmonic accompaniment is never strictly “wrong” or “right.” In this work, we use chord choices and their progressions as proxies for harmonic movement. By using the information contained in a chord at a given moment and an input melody note, this system will output what accompanying chord could be used next in a musical passage. An additional parameter can be tweaked which determines how “conservative” the next chord choice would be, providing more possibilities for harmonization. Composers and songwriters can use this tool to brainstorm new harmonic choices, and anyone with an interest in music can find new chord changes to their favorite songs given that they have the melody notes for such a song. This work has applications in music generation and symbolic music representation.

1. INTRODUCTION
Music harmonization is a longstanding area of research in music generation [1]. Some automatic harmonization systems in the past have used Markov models [2, 3], but these models were informed by training data and only prescribed one harmonization output based on the inputs. Other work has used statistical inputs in conjunction with general rules of harmony to automatically create harmonizations [4]. [5] exploits part-whole hierarchies for automatic harmonization, and others have employed deep learning to harmonize using triadic approaches [6] or template methods [7].

This system uniquely blends concepts from symbolic processing and music theory, providing musicians and laypeople alike a wellspring of interesting new harmonic progressions and accompaniments for melodies. By showing a method of music generation that is informed by context-independent voice-leading principles,
Table 1. Chord distance based on their note interval difference. The B in Cmaj7 is one semitone away from Bb in C7, and the distance between Cmaj7 and Dmaj7 is 6: C voice-leads to C# (1 half step), E travels to D (2 half steps), G goes to F# (1 half step), and B transitions to A (2 half steps).

<table>
<thead>
<tr>
<th>Chord1</th>
<th>Chord2</th>
<th>d(chord1, chord2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cmaj7</td>
<td>Cmaj7</td>
<td>0</td>
</tr>
<tr>
<td>Cmaj7</td>
<td>C7</td>
<td>1</td>
</tr>
<tr>
<td>Cmaj7</td>
<td>Dmaj7</td>
<td>6</td>
</tr>
</tbody>
</table>

Table 2. Some tetrachords and their spellings. When a chord contains more than four notes, the 5th of the chord is removed (as in Fmin9). In the case of a triad, the tetrachord is doubled at the root of the chord. In order to transition to a tetrachord, it must have at least one note that matches the melody note input.

<table>
<thead>
<tr>
<th>Tetrachord</th>
<th>Cmaj7</th>
<th>Fmin9</th>
<th>Dmin6</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Note 1</td>
<td>C</td>
<td>F</td>
<td>D</td>
<td>C</td>
</tr>
<tr>
<td>Note 2</td>
<td>E</td>
<td>Ab</td>
<td>F</td>
<td>E</td>
</tr>
<tr>
<td>Note 3</td>
<td>G</td>
<td>Eb</td>
<td>A</td>
<td>G</td>
</tr>
<tr>
<td>Note 4</td>
<td>B</td>
<td>G</td>
<td>B</td>
<td>C</td>
</tr>
</tbody>
</table>

3. METHODS

3.1 Choosing Chords

3.1.1 Neo-Riemannian Voice-Leading

In this work, we use the context-independent Neo-Riemannian voice-leading system [13] to determine how similar two chords are to one another. In this system, two chords are close to each other if only a few semitonal shifts are required to transform one chord into the other [8]. We define the Neo-Riemannian distance $d(\cdot, \cdot)$ as the number of total semitonal shifts required to change one chord into another and use this metric determine how the output harmony evolves. See Table 1 for examples.

3.1.2 Chord Transitions

Inspired by [2], we use a Markov model to determine how to transition from one chord to another based on these neo-Riemannian distances.

In order to do this, we first construct a set of tetrachords, or four-note chords, for every natural melodic note (seven sets total—one for A, B, C,..., G). See Table 2 for some tetrachords we used and their spellings. We construct stochastic probabilities of transitioning from chord$_1$ to chord$_2$, given by:

$$p(chord_2|chord_1) = \frac{(d(chord_1, chord_2) + 1)^{-t}}{\sum_c(d(chord_1, chord_c) + 1)^t} \quad (1)$$

where $t$ is the parameter defined by $e^{-\text{temperature}}$, where temperature is a user-inputted value between -5 and 5, $c$ is the set of all chords containing the future melody (note the denominator of Eqn (1) is a normalizing factor), and $d$ is the Neo-Riemannian distance.

3.2 Temperature Tuning

The temperature of the system can take on float values between -5 and 5. The value $t$ is defined as $e^{-\text{temperature}}$. When the temperature is -5, the probability of transitioning to a chord that is dissimilar to the previous chord is very small, as large distances are inflated by a large $t$, bringing down the transition’s probability. This corresponds to a system that conservatively chooses chords. When the temperature is 5, the probability of transitioning to a chord that is extremely different from the current chord is much higher: in this case, every possible chord transition probability is about the same, because even the large-distance transitions are brought close to 1 after being exponentiated with a small, positive $t$. In this way, one set of melodic notes can create vastly different harmonizations based on the temperature of the model.

4. LIMITATIONS

There are several limitations to this work. In Figure 1, for example, Harmony 3 is consonant given the melodic inputs, but it is dissonant when accounting for the melodic notes that are not included in the input. Here, Abmaj7 is a valid chord choice for the melodic note of G, but the melodic note A is dissonant given the Abmaj7 chord. A future iteration of this system will allow for multiple melodic inputs for each output chord a user requests, and each output will have to accommodate all melody notes between chord transitions, not just the first one in a melodic line.

Furthermore, only natural notes (A, B, C,...,G) can be used as melodic input to our system. This was to ensure that the melodies inputted were only in the key of C major or A minor, allowing for easier construction of the set of chords that could harmonize the melodic notes. While harmonic accompaniments can contain notes or even roots that are non-natural (Emin9, for example, has an F#), a user must sometimes transpose an input melody into the proper key in order to use our system. Further work will be done to expand the capabilities of this system so as to accommodate non-natural melodic note inputs.

5. CONCLUSION

Harmonization is an paramount to any comprehensive music generation system. In this work, we create possible harmonic accompaniments of input melodies. We also outfit our system with a parameter that dictates how “conservative” or “unlikely” a chord transition output will be. This tool can be found at https://github.com/timothydgreer/4_part_harmonizer and can be used by musicians to create accompaniment for their melodies or more generally anyone with a curiosity in music generation and/or symbolic processing.

\(^2\) These harmonizations may indeed inspire new non-natural, melodic compositions as well.
6. REFERENCES


