Ergonomic fingerings for guitar chord sequences

DD143X Degree project in Computer Science

School of Computer Science and Communication (CSC)
KTH Royal Institute of Technology
Supervisor: Anders Askenfelt

Abstract

An algorithm has been developed which generates fingerings for guitar chord sequences using a cost minimizing function. The performance of the algorithm was evaluated by two professional guitarists and the results indicate that the approach gives viable solutions. With a few minor improvements the algorithm can be a good way to calculate sets of guitar chord fingerings that suit specific needs concerning playability and ergonomic constraints.

Sammanfattning

En algoritm har utvecklats som genererar fingersättningar för ackordsekvenser på gitarr med hjälp av en kostnadsminimerande funktion. Utvärderingen av algoritmen gjordes av två professionella gitarrister och resultaten indikerar att denna approach ger dugliga resultat. Efter ett par smärre förbättringar kan algoritmen vara ett bra sätt att beräkna fingersättningar som uppfyller specifika krav på spelbarhet och ergonomiska hänsyn

Christoffer Dahlgren
Hjortronvägen 15
196 35 KUNGSÄNGEN
073-846 06 80
chd@kth.se

Sebastian Grahn
Professorsslingan 41
114 17 STOCKHOLM
070-480 91 70
segrahn@kth.se

Table of contents

Abstract 1

Sammanfattning 1

Introduction 4

Problem Statement 4

Scope 5

Background 6

Basic guitar structure 6

Literature review 6

Method 7

Algorithm 7

Penalties and Weighting 7

Software structure 8

Main 8

Note and Chord 8

NoteAnalyzer 8

CalcChords 8

CalcCost 9

Evaluation 9

Results 11

Discussion 13

Conclusions 15

Expansion & further research 16

References 17

Appendix A 18

On the guitar chord and the concepts of its construction 18

Inversion of triads 19

Appendix B 20

Algorithm in short 20

Appendix C 21

Raw data 21

First chord sequence 21

Second chord sequence 21

Third chord sequence 21

Fourth chord sequence 22
Total and average scores 22
Generated sets of fingerings 23
A brief introduction to tablature notation 23
First set 23
Weighting A 23
Weighting B 23
Weighting C 23
Second set 24
Third set 24
Fourth set 25

Introduction

The modern guitar is an instrument with roots dating back as far as the 12th century evolving into its current shape sometime in the beginning of the 19th century. It is now commonly used in a wide variety of musical genres, ranging from classical music to heavy metal. The popularity makes it a common first instrument to learn, especially with its wide use in western popular music.

The guitar shares a common feature of all plucked and bowed string instruments, namely that the same note (pitch) can be played on different strings at different positions along the neck. This allows for a plethora of options for playing a single note, and a large number of ways to combine them.

The purpose of this study concerns ways of finding efficient fingering on the guitar for a given chord sequence. Such an application would fill a need for guitar players who want to find a new way of calculating chords, specifically in order to stay within a limited range of the fret board. The exploration of possible solutions is based on professional guitarists preferences and experiences regarding playing technique and ergonomic standpoints. An efficient fingering is interesting both from an ergonomic as well as an ease of play standpoint.

Problem Statement

The specific problem investigated in this study is if it is possible to generate fingerings for relatively simple guitar chord sequences automatically with primary concern to ergonomics. A further question if such automatically generated fingerings are reasonably easy to play. The goal of the study is a program that can generate a set of viable fingerings for a given chord sequence restricted to a specified part of the fret board.

Scope

The complexity and variation of fingerings for a guitar chord sequence grows fast with the length of the chord sequence. Even for relatively short sequences a limited scope is required. According to interviews with professional guitar players, the design of the guitar disqualifies a wide range of theoretically possible variations. This is due to the relatively long distance from the first to the last fret, making long movements of the position of the left hand highly impractical to play from a timing perspective. This study concerns sets of fingerings with primary respect to ergonomic constraints. Musical considerations are beyond the scope of the study.

A basic limitation made was to specify a range of fingering positions, corresponding to a certain number of frets, to focus on. The motivation was to narrow down the possible ways of playing a chord to a reasonable number. The limitation of the playing range on the fret board was decided to be 4 frets, plus one extra fret in either direction. To further limit the amount of chord fingerings available, only chords with three notes (triads in root position and first and second inversions, see Appendix A) were taken into consideration.

Background

Basic guitar structure

The regular guitar of today is fitted with six strings with a standard tuning E4, B3, G3, D3, A2 and E2, from top to bottom string. The corresponding range in fundamental frequency covered by the open strings is 82 – 330 Hz. A certain note (pitch) is played by selecting an appropriate string and shorten the string length accordingly. The fingerboard has about 18 (depending on the type of guitar) metal strips (frets), which terminates the speaking length of the string as the string is pressed down by the player's finger at the 'passive string' side of the fret. The difference in pitch between two frets is by definition a semitone corresponding to one step on the chromatic scale (see Appendix A).

Literature review

A literature review was made which revealed important facts about the use of different versions of chords (see Appendix A). The findings of Hubbard and Datteri that a major chord and its first and second inversion in different positions may be harmonically equivalent [3] were of great value for the design of the algorithm, indicating that they are interchangeable.

In general, the complexity and physiological strain of fingering a chord is dependent on many factors, which influence the design of the algorithm (see Sect. 4 - Algorithm). An important observation based on conversations with professional guitarists was made by Miura et al., who concluded that "increasing the amount of movement of the left wrist" has a greater impact on the performance overall compared to "increasing the amount of stretch in the fingers of the left hand"[2].

Supporting the idea that playing a chord sequence is indeed a cost problem, Heljink and Meulenbroek reached the conclusion: "The problem of selecting optimal locations and fingers for every note in a sequence can indeed be seen as the planning of a low-cost series of postures that satisfy relevant task constraints"[1].

Method

An algorithm was designed that could generate several possible fingering versions for each chord. This made it possible to compare different combinations of chord fingerings using different weights for transitions (hand movement along the fingerboard and change of fingers to another string) and the actual difficulty of a static fingering of a chord. This design method was guided by an interview of a professional guitarist[1] as well as the findings of the literature review.

Algorithm

The objective of the algorithm was to generate a fingering for a chord sequence, taking ergonomic concerns into account. This was done by transforming the problem into a cost-minimizing problem with different penalties given to specified hand movements and finger straining.

Penalties and Weighting

The algorithm was based on a small set of rules causing the following hand movements and finger distances to incur a penalty:

1. Moving the hand in parallel to the fret board (in the string direction)
2. Moving the hand perpendicular to the fret board (sideways across the width of the neck)
3. The distance between fretted (pressed) strings in a chord (sideways across the width of the neck
4. The distance between frets for different strings in a chord (in the string direction)

Three different sets of weightings were used:

A. A reference weighting with all weights equal
B. A high cost for moving the hand, based on the literature and the interview
C. A high cost for stretching. In contrast to B, moving the hand incurs a low penalty

The lowest cost combination for each of the weightings was calculated and produced a distinctly different set of fingerings for all the chord sequences evaluated.

[1] Ola Bengtsson, lector in electric guitar and head of the institute of jazz, at The Royal

Software structure

The algorithm was implemented in six processing steps: Main, NoteAnalyzer, CalcChords, CalcCost, Chord and Note (see Fig. 1).

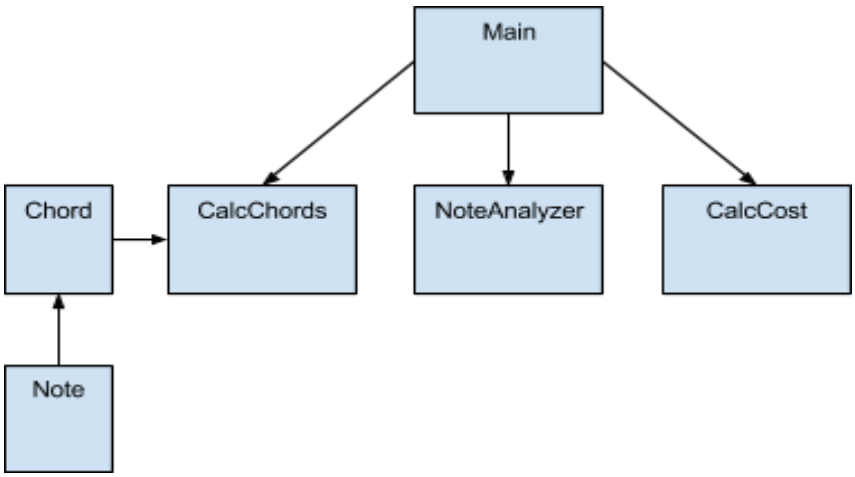

Figure 1: Illustration of the structure of the algorithm

Main

The Main step controls the instances of the three other processes CalcChords, NoteAnalyzer and CalcCost and makes the call to start their processes.

Note and Chord

The two help processes Note and Chord create objects representing notes and chords. The note object contains three integers, representing respectively a note, a string and a fret. The chord object can be instantiated with either a set of three integers or three note objects (the latter of which is the representation of a complete triad).

NoteAnalyzer

The NoteAnalyzer process handles the building of scales (represented by integers), calculated according to classic scale progression. It then carries out the first tasks: given one chord (in the form X, Xmin, Xdim, Xaug, Xsus2, Xsus4, where X is the scale) it parses this and splits it up into two parts, the scale and the type of triad. The scale part is analyzed and properly connected to one of the generated scales, and a method is called with the scale and the triad type as parameters, which (with the formulas in ill. 2 as a guide) calculates the corresponding notes (in integers) and instantiates a temporary Chord object without fret and string variables with the notes as input. This process is repeated for each chord input.

CalcChords

The CalcChords process takes the chord object from NoteAnalyzer together with a user-defined range of frets as input. It starts by calculating all of the valid notes from the chord in the defined range of the fret board. With these notes a method calculates all the possible triads where it can find the three notes on different strings without breaking the rule of max fret width for a chord. When it has found all of the chords they are put into a list and returned.

CalcCost

Considering that the problem is divisible (each transfer between chords and fingering of each chord is, in itself, a cost minimizing problem) into smaller parts, the CalcCost process takes a dynamic programming[2] approach to solve the cost problem.

At first it calculates the shortest path between **all** the versions of the first chord in the given chord sequence to **all** the versions of the second chord, and for each of the versions of the second chord, it saves the path with the lowest cost, now “forgetting” the first chord regarding the cost and solely keeping track on the total cost to get to respective version of the latest chord.

This is done sequentially with all chords until it reaches the last chord, where it calculates the lowest cost from each of the versions of the last chord. A comparison of all the total costs is now executed, saving the lowest, and the optimal path is calculated based on this saved lowest cost.
A pseudo code version of the algorithm is found in appendix B.

Evaluation

Two experienced guitar players[3] evaluated four chord sequences of increasing length and complexity

Am, F, C, G	(in fret range 6-9)
Em, G, D, Am	(in fret range 6-9)
G, C, G, D, C, C	(in fret range 5-8)
Am, Dm, G, C, F, Dm, G, E	(in fret range 1-4)

Based on this evaluation the performance of the fingering algorithm as well as the different weightings was analyzed. The evaluation was conducted in the following way.

Both guitarists were presented with three separate sets of generated fingerings (not knowing which fingering was generated from which weighting) and played them a couple of times. They were asked to grade two statements, "How natural did the fingering feel?" and "How strenuous did the fingering feel?", from 1 to 5 (1 being "very unnatural" and "very strenuous" respectively, and 5 being "very natural" and "not at all strenuous" respectively).

They were also given two multiple-choice questions:
1. "Which of these three fingerings would you use from an ergonomic standpoint?"
2. "Which of these three fingerings would you use from a musical standpoint?"

The last statement was beyond the scope of this study, but could be interesting for future development of the algorithm.

---

[2] "Dynamic Programming." *Wikipedia*. Wikimedia Foundation, 04 Oct. 2014. Web. 11 Apr. 2014. <http://en.wikipedia.org/wiki/Dynamic_programming>.
[3] The aforementioned Ola Bengtsson, and one of his students named Emmanuel Hailemariam

Finally, a free form discussion was held, where details concerning the weightings and principles behind the software were revealed and the opportunity was given for one of the guitarists to comment on what he felt was missing in the fingerings, and which parameters, if any, were to be adjusted to achieve a desired result.

Results

The results from the evaluation are presented in figures 2 to 5
The raw data are included in Appendix B.

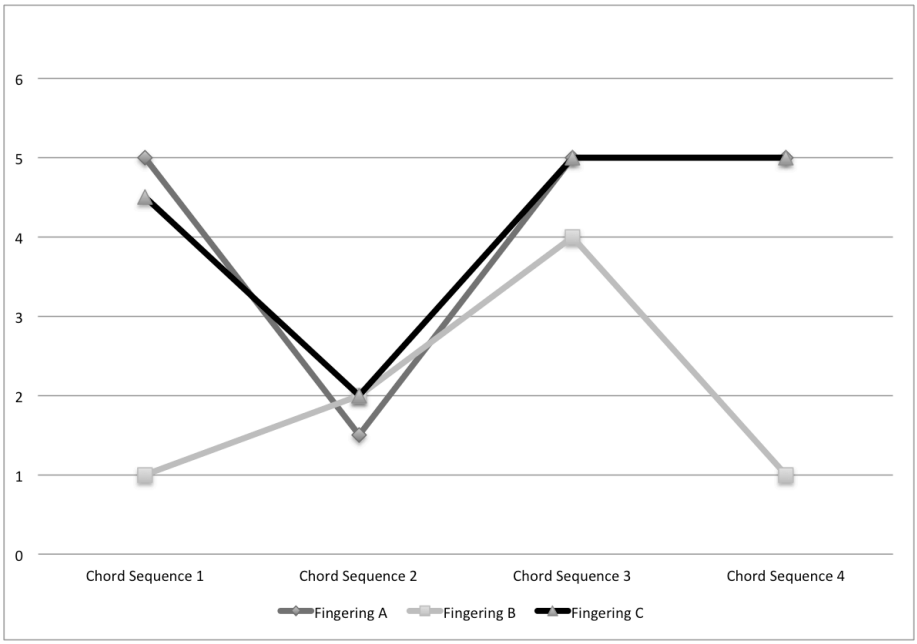

Figure 2: Average grade of the statement "How natural does this fingering feel?" 5 being very natural, 1 being very unnatural.

Figure 2 shows the average of the scoring on question 1. "Which of these three fingerings would you use from an ergonomic standpoint?" for the different weighting sets, graded between 1 and 5. Note the difference between fingering B and the other two.

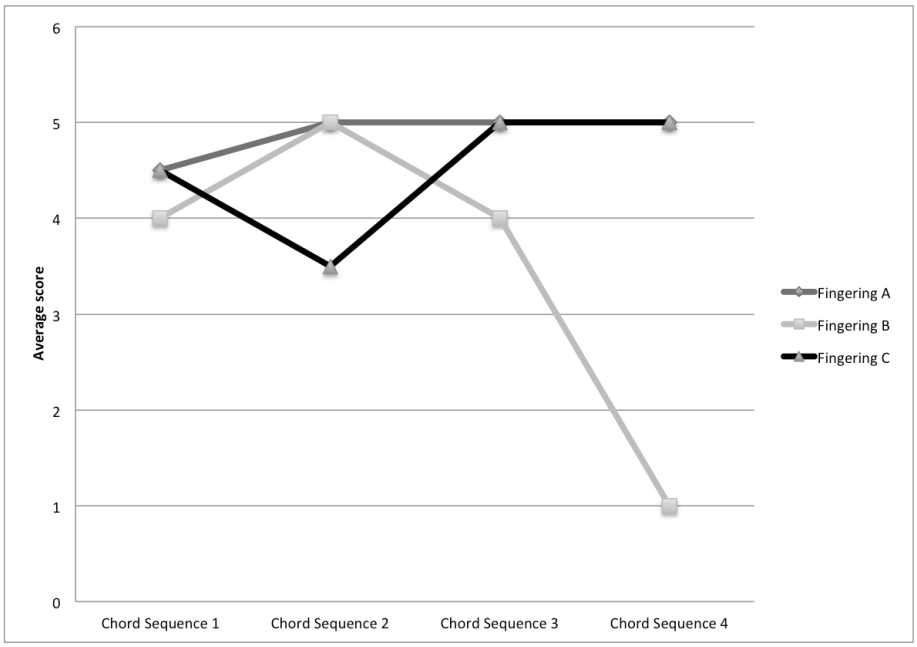

Figure 3: Average grade of the statement "How strenuous was this fingering?" 1 being very strenuous and 5 being not at all strenuous

Figure 3 shows, for respective chord sequences, the average scoring on the strenuous statement from both guitarists for the different sets of fingerings, graded from 1 to 5. Note again the difference between fingering B and the other two.

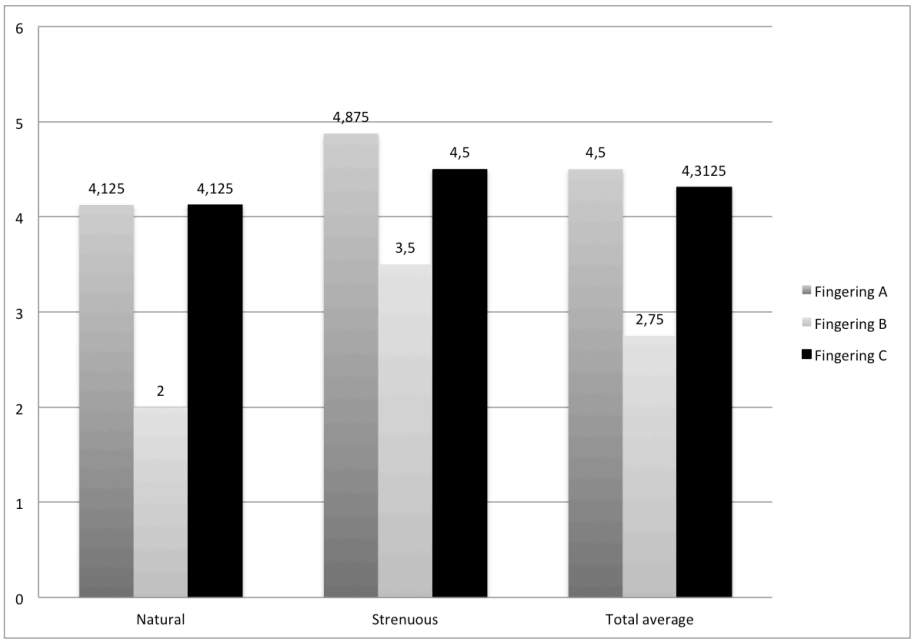

Figure 4: Average score per category and total average, 1 to 5, where 5 is the best possible mark

Figure 4 shows the average score per category, and the total average of both categories for each set of fingerings. Interesting to note is the difference between fingering B and the other two. The total average scores of the different generated fingerings are: (A) 4.5 points, (B) 2.8 points and (C) 4.3 points.

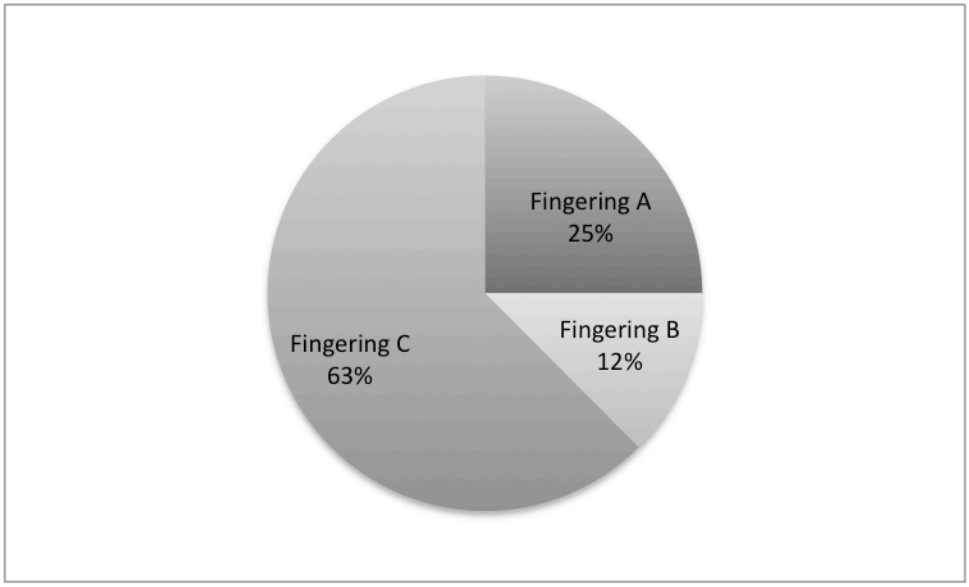

Figure 5: Chosen fingerings in total

Figure 5 shows what percentage of the given fingerings was chosen with respect to the ergonomics of the fingerings.

Discussion

As noted in the result, the average scores of the different generated fingerings are: (A) 4.5 points, (B) 2.8 points and (C) 4.3 points.

The results show a small advantage for fingering A, with C not far behind. A significantly lower score for the B fingering was observed, partly due to a few outliers for chord sequence 1 and 4 in particular. A more surprising result was that fingering C ergonomically in 5 out of 8 cases, in spite of the fact that it was graded lower, however, only slightly. This might be due to a minor methodological flaw, namely that the order of the fingerings were not randomized causing the C set of fingerings to always be the last one played, which possibly can have generated a small bias.

Also worth noting is that, with a few exceptions, none of the sets of fingerings were considered to be particularly strenuous to play. This is probably partly due to the fact that both of the evaluating guitar players were highly professional and therefore very proficient in executing all manners of fingerings. The result may also indicate that the software in fact does work in exactly the way desired to generate sets of fingerings that rarely result in any specifically complicated or strenuous versions.

The lowest scores, both on average and individually, were recorded for the B sets of fingerings. This was not in line with expectations, since these sets were generated according to both what was found in the literature and what the guitarists stated. The literature was not entirely clear on whether moving of the hand was primarily negatively affecting performance from a timing perspective (which would be beyond the scope of this study) or from ergonomic standpoints. However, the results suggest that the ergonomics of moving the hand are indeed a factor for the strain and perceived naturalness of the sets of fingerings.
With minor adjustments, the low cost approach to calculating a chord sequence both ergonomically viable and with a relatively natural feel seems to be a valid solution to the problem (however some of the weightings seem to pose a problem). It would also be easy to adjust the software to handle other chords than triads.

The A weighting (equal weights) gave the best score and the worst scored was the B- weighting. A probable cause for this result is the way the penalty for switching frets is calculated. In the current implementation we did not differentiate between calculating the width (i.e. stretching of the fingers) of a chord and the movement of the hand.
The example illustrated below [see Fig. 5] describes the problem. The cost from the first chord to the second chord would be 3 (1+1+1=3), as the user would need to move the hand one fret together with no stretch at all. The cost from the second chord to the third chord would also be 3 (0+3+0=3), as the user would not need to move the hand at all, but it would require stretching the hand over four frets (from the fifth fret to the eighth fret).

A possible solution to this problem is suggested in section 8 - “Expansion & further research”. In short, instead of calculating the total of the movements for each finger another way would be to keep track on both the width of a chord and the longest movement between chords.

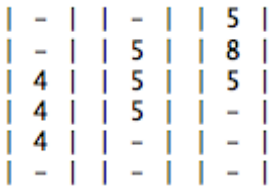

Figure 6: An example chord (for an explanation on tablature, refer to Appendix B – A brief introduction to tablature notation)

Conclusions

We have proven that it is possible to generate chord fingerings for simple chord sequences automatically. Different ergonomic considerations would most of the time generate different chord sequences, which was one of our goals as well.

As shown earlier in Results, almost every suggested chord sequence got a good grade in simplicity, hence it seems we have also been able to eliminate the more difficult versions of chord sequences.

Therefore we draw the conclusion that the program is useful in its current state of development for guitar players who want to think outside the box regarding their guitar playing and avoid the most common variants of chords. However for the program to be really useful in a more professional setting, minor changes need to be implemented, discussed in more detail in Expansion & further research.

Expansion & further research

The algorithm can be refined in several respects. A necessary change needed would be to implement a better cost function handling the problem discussed at the end of *Conclusions.* The algorithm needs to differentiate between the width of a chord and the total movement between chords. This could be done by allocating separate variables for the width and the actual movement of the hand between two chords, and take both these variables, multiplied by different weightings, into account when calculating the cost.

The free form discussion with the guitarists concerning the structure and weightings of the program opened up for several suggestions of how to improve and expand the program. The primary issue was to add a mechanism for ensuring that the chord sequences have voice leading as a criterion for generation of fingering sets. This would not be a difficult implementation. The criterion is to add a maximum range for the total number of notes a chord is allowed to span (not more than 15 semi tones). A result of this will likely be that the sets of fingerings that meet this criterion will be fewer than what the algorithm generates at present. This would give fewer options to “optimize” sets than currently, which might streamline the sets to be more equal despite different weightings. However, in a broader and more useful context, this may not be as detrimental as it seems at first glance, especially if it was decided to use only the best (or best and second best) weighting(s).

References

1. Heijink, Hank, and Ruud G. J. Meulenbroek. "On the Complexity of Classical Guitar Playing: Functional Adaptations to Task Constraints." *Journal of Motor Behavior* 34.4 (2002): 339-51. Print.
2. Miura, Masanobu, Isao Hirota, Nobuhiko Hama, and Masazo Yanagida. "Constructing a System for Finger-position Determination and Tablature Generation for Playing Melodies on Guitars." *Systems and Computers in Japan* 35.6 (2004): 10-19. Print.
3. Timothy L. Hubbard and Darcee L. Datteri ”Recognizing the Component Tones of a Major Chord” *The American Journal of Psychology* Vol. 114, No. 4 (Winter, 2001), pp. 569-589

Appendix A

On the guitar chord and the concepts of its construction

A guitar chord is a set of notes played either simultaneously or sequentially (called arpeggio), built around the twelve notes in the octave (illustrated in the chromatic circle [Figure 7], where the distance between two consecutive notes are exactly one semi note)[4].

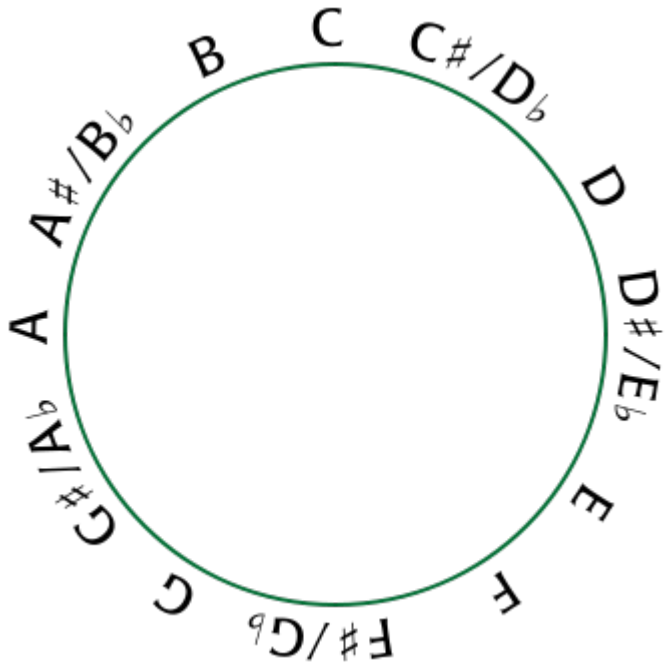

Figure 7: The chromatic circle

This paper is focused on what is commonly known as triads. A triad is a chord consisting of three notes, the **root** note (C), the **third** (which is a major third above the root in the case of the major triad or a minor third above the root in case of the minor triad. Other varieties occur in other chords) and the **fifth** (a perfect fifth over the root and, therefore, a minor third above a major third or a major third above a minor third). Similar patterns can be used across all scales and for all triads, as indicated (for chords in E) [Figure 8].

[4] "Guitar Chord." *Wikipedia*. Wikimedia Foundation, 04 Oct. 2014. Web. 11 Apr. 2014. <http://en.wikipedia.org/wiki/Guitar_chord>.

Triad	Formula (note nr. in desired scale)	Notes
E major	1, 3, 5	E, G#, B
E minor	1, b3, 5	E, G, B
E diminished	1, b3, b5	E, G, A#
E augmented	1, 3, #5	E, G#, C
E sus4	1, 4, 5	E, A#, B
E sus2	1, 2, 5	E, F, B

Figure 8: Formulas for calculating triads (a b together with a numbered scale note stands for the note, out of all notes, before the note in the current scale, # together with a numbered scale note stands for the note, out of all notes, after the scale note)

These formulas give a way to find the triads for each scale[5].

Inversion of triads

As mentioned above the triads consist of three notes; the root, the third and the fifth. In the root position of a chord the notes occur in that order. For the first inversion of a chord the root note is moved to the last position in the chord which results in the note order third, fifth and root. Using the first inversion the second inversion of a chord is achieved by moving the third note to the last position of the chord which results in the note order fifth, root, third.

The three mentioned structures are displayed in the illustration below [Figure 9].

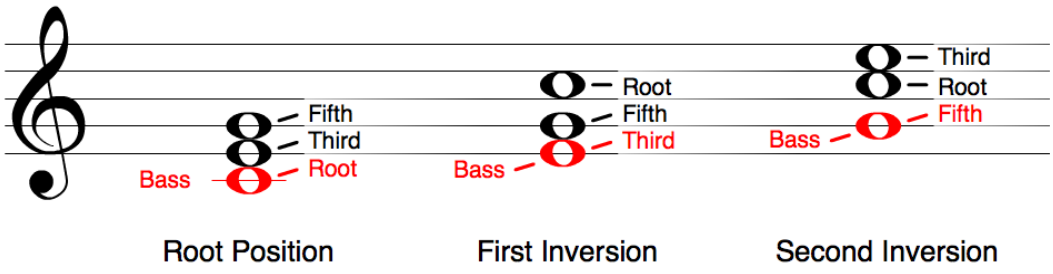

Figure 9: Root position, first & second inversions[6]

[5] "Understanding Guitar Triads." Web. 11 Apr. 2014. <http://www.ultimate-guitar.com/lessons/chords/understanding_guitar_triads.html>.

[6] "Mandolin Theory – Understanding Chords Inversions." *The Mandolin Tuner*. Web. 11 Apr. 2014. <http://www.themandolintuner.com/chords-inversions/>.

Appendix B

Algorithm in short

This is the algorithm for calculating chords in pseudo code.

Input: A chord sequence as a string and a specific range of the neck
1: For each chord
2: Find the notes of the chord
3: Find these notes position on the neck in our range
4: Create valid combinations of the different notes as a list of chord versions
5: For the first chord list
6: Calculate the grabbing cost for each chord
7: Add cost to the matrix position for costToGetHere
8: For the remaining chord lists
9: For each chord version in this list (Chord *to*)
10: Calculate the grabbing cost
11: For each chord in the last list (Chord *from*)
12: Calculate the cost for moving between *from* and *to*.
13: Add the *from* which results in the lowest total cost of row 12 together with the value from costToGetHere(from) to the costToGetHere(to) and also add the from chord to a list bestPath
14: Print the list bestPath as the result

Appendix C

Raw data

First chord sequence

Fingering	Statement	Scoring	Avg. scoring
A	Unnatural/natural	5, 5	5
	Strenuous/not strenuous	4, 5	4.5
B	Unnatural/natural	1, 1	1
	Strenuous/not strenuous	4, 4	4
C	Unnatural/natural	5, 4	4.5
	Strenuous/not strenuous	5, 4	4.5
	Which preferred ergonomically?	C, A	N/A
	Which preferred musically?	C, C	N/A

Second chord sequence

Fingering	Statement	Scoring	Avg. scoring
A	Unnatural/natural	1, 2	1.5
	Strenuous/not strenuous	5, 5	5
B	Unnatural/natural	1, 3	2
	Strenuous/not strenuous	5, 5	5
C	Unnatural/natural	1, 3	2
	Strenuous/not strenuous	4, 3	3.5
	Which preferred ergonomically?	A, B	
	Which preferred musically?	-, C	

Third chord sequence

Fingering	Statement	Scoring	Avg. scoring
A	Unnatural/natural	5, 5	5

	Strenuous/not strenuous	5, 5	5
B	Unnatural/natural	4, 4	4
	Strenuous/not strenuous	4, 4	4
C	Unnatural/natural	5, 5	5
	Strenuous/not strenuous	5, 5	5
	Which preferred ergonomically?	C, C	N/A
	Which preferred musically?	C, C	N/A

Fourth chord sequence

Fingering	Statement	Scoring	Avg. scoring
A	Unnatural/natural	5, 5	5
	Strenuous/not strenuous	5, 5	5
B	Unnatural/natural	1, 1	1
	Strenuous/not strenuous	1, 1	1
C	Unnatural/natural	5, 5	5
	Strenuous/not strenuous	5, 5	5
	Which preferred ergonomically?	C, C	N/A
	Which preferred musically?	C, C	N/A

Total and average scores

Fingering	Statement	Total score	Total avg. scoring	Total avg. both categories
A	Unnatural/natural	33	4.125	4.5
	Strenuous/not strenuous	39	4.875	
B	Unnatural/natural	16	2	2.75
	Strenuous/not strenuous	28	3.5	
C	Unnatural/natural	33	4.125	4.3125
	Strenuous/not strenuous	36	4.5	

Generated sets of fingerings

These are the generated sets of fingerings for the chords used. The three sets each correspond to a different weighting according to the list on page 6.

A brief introduction to tablature notation

The sets of fingerings are generated in what is known as tablature. It indicates which string should be pressed at which fret (i.e., for the first set below, the first chord is pressing strings 2, 3 and 4, counted from the bottom, pressed down at fret 10, 9 and 10 respectively, the second chord strings 2, 3 and 4 pressed down at fret 10, 10, 10 respectively and so on. Each column represents one chord in the sequence).

First set

Weighting A

```
| - | | - | | - | | - |
| 10| | 10| | - | | - |
| 9 | | 10| | 9 | | - |
| 10| | 10| | 10| | 9 |
| - | | - | | 10| | 10|
| - | | - | | - | | 10|
```

Weighting B

```
| 8 | | - | | - | | 7 |
| 5 | | - | | - | | 8 |
| - | | 5 | | - | | - |
| 7 | | 7 | | 5 | | - |
| - | | 8 | | 7 | | 5 |
| - | | - | | 8 | | - |
```

Weighting C

```
| 5 | | 5 | | - | | 7 |
| 5 | | 6 | | 5 | | - |
| 5 | | 5 | | 5 | | 7 |
| - | | - | | 5 | | 5 |
| - | | - | | - | | - |
| - | | - | | - | | - |
```

Second set

-		10		-		-
-		-		10		10
9		-		-		9
9		9		-		10
10		10		9		-
-		-		10		-

-		7		-		-
-		-		7		-
-		7		-		5
5		5		7		7
7		-		5		7
7		-		-		-

7		7		-		8
8		8		7		-
-		7		7		-
-		-		7		7
7		-		-		7
-		-		-		-

Third set

-		-		-		-		-		-
-		5		-		-		5		5
4		5		4		-		5		5
5		5		5		4		5		5
5		-		5		5		-		-
-		-		-		5		-		-

-		-		-		5		-		-
-		-		-		7		-		-
-		5		-		-		5		5
5		5		5		-		5		5
5		7		5		5		7		7
7		-		7		-		-		-

-		-		7		-		-		-
-		5		8		7		5		5
4		5		7		7		5		5
5		5		-		7		5		5
5		-		-		-		-		-
-		-		-		-		-		-

Fourth set

-		-		-		-		-		-		-		-
-		3		3		-		-		3		3		-
2		2		-		-		2		2		-		-
2		3		-		2		3		3		-		2
3		-		2		3		3		-		2		2
-		-		3		3		-		-		3		4

5		-		-		-		-		-		3		-
1		3		3		-		-		3		3		5
-		2		-		-		2		2		-		1
2		3		-		2		3		3		-		-
-		-		2		3		3		-		2		2
-		-		3		3		-		-		-		-

5		-		-		0		1		-		3		4
5		3		0		1		1		3		3		5
5		2		0		0		2		2		4		4
-		3		0		-		-		3		-		-
-		-		-		-		-		-		-		-
-		-		-		-		-		-		-		-