

# A GENETIC ALGORITHM FOR THE AUTOMATIC GENERATION OF PLAYABLE GUITAR TABLATURE

*D.R. Tuohy and W.D. Potter*  
Artificial Intelligence Center  
University of Georgia  
Athens, Georgia USA

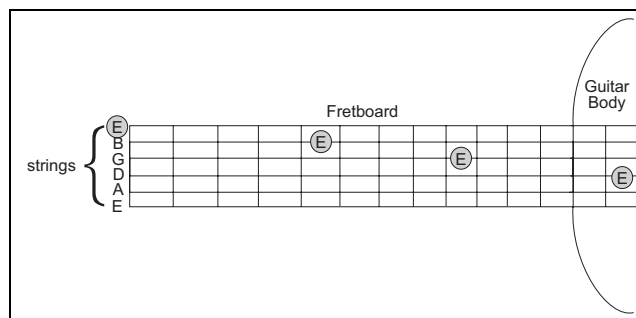
## ABSTRACT

This paper describes a method for mapping a sequence of notes to a set of guitar fretboard positions (tablature). The method uses a Genetic Algorithm (GA) to find playable tablature through the use of a fitness function that assesses the playability of a given set of fretboard positions. Tests of the algorithm on a variety of compositions demonstrate an excellent ability of the GA to discover easily playable tablature that maintains a high degree of consistency with published tablatures transcribed by humans. The algorithm was also found to generally outperform commercial software designed for the same purpose. We conclude that the GA can reliably produce good tablature for any piece of guitar music.

## 1. INTRODUCTION

Stringed instruments often require a great deal of experience and decision making on the part of the performer. A given note on the guitar may have as many as five different positions on the fretboard on which it can be produced. Figure 1 shows a guitar fretboard, and four different fretboard positions on which the same “E” can be played. To play a piece of music, the performer must decide upon a sequence of fretboard positions that minimize the mechanical difficulty of the piece to at least the point where it is physically possible to be played. This process is time-consuming and especially difficult for novice and intermediate players and, as a result, the task of reading music from a page as a pianist would is limited only to very advanced guitar players. To address this problem, a musical notation known as tablature is used. Tablature describes to the performer exactly how a piece of music is to be played by graphically representing the six guitar strings and labeling them with the corresponding frets for each note, in order.

The goal of our research is to automatically discover very good tablature for any piece of music. In the next section we will describe recent commercial and academic attempts to accomplish the same feat. We will then describe our genetic algorithm approach and offer an explanation for why it is better suited for this problem



**Figure 1.** Four different fretboard positions for the 3rd octave “E”.

## 2. BACKGROUND

### 2.1. Commercial Software

Several programs on the market today convert music into tablature. These programs are, however, notorious for producing unplayable or unnecessarily difficult tablatures[6]. This is because they often depend too heavily upon rules of music theory relating to harmony and guitar composition[12]. For example, “open” strings are considered to be easier to play on guitar because they do not require depressing the string on a fret. However, it is often easier for a note to be played on a depressed string if that string happens to already be depressed at that point in the music. Software that we tested seems to take the inadvisable approach of creating tablature note by note, rather than devising a strategy for the piece as a whole. This approach necessarily results in poor tablature for many pieces of music. For these reasons, current commercial software often depends upon the user to edit a tablature after it has been generated[12].

### 2.2. Academic Research

A method for discovering playable tablature was described by Samir I. Sayegh[11]. His “optimum path paradigm” describes a sequence of fretboard positions as a sequence of hand states, the goal being to find the optimum path through the states. More recently, there have been algorithms that build upon Sayegh’s approach at the University of Torino[9] and the University of Victoria[10]. Both approaches have produced excellent tablatures for the pieces

on which they were tested, but both have limitations. The prototype from the University of Torino cannot account for situations where more than one note needs to be played simultaneously (a chord). The program from the University of Victoria appears to be a “lazy learner” which improves its accuracy by requiring customized learning for each piece. Our algorithm aims to be able to generalize to any piece with a satisfactory degree of accuracy, though it will be difficult to assess which algorithm produces more desirable results. Another group from Doshisha University reports success in generating tablature superior to that of commercial software, but once more the program is limited to producing tablature for melodies without chords[6]. It should be noted that the former two approaches include Left Handed Fingering notation, which instructs the player on which specific fingers to use. Our approach does not currently support this feature.

### 3. THE GENETIC ALGORITHM APPROACH FOR TABLATURE GENERATION

#### 3.1. The Genetic Algorithm

A Genetic Algorithm (GA)[7][8] is a stochastic search algorithm that aims to find good solutions to difficult problems by applying the principles of evolutionary biology. Evolutionary concepts such as natural selection, mutation and crossover are applied to a “population” of solutions in order to evolve a very good solution. Problems suited for a GA are those whose number of possible solutions (search space) is so large that finding good solutions by exhaustive search techniques is computationally impractical. The genetic algorithm is also useful for tackling search spaces with many scattered maxima and minima. This property of GAs is essential in the search for tablature, as the search space can be quite large, generally on the order of  $3^n$  where  $n$  is the number of notes.

#### 3.2. Implementation

The population for our GA is a collection of tablatures that are valid, though not necessarily desirable, for a given piece of music. The initial population is generated randomly. A tablature “chromosome” is defined as a sequence of chords. A chord is a “gene” and consists of fretboard positions for all the notes in that chord. A chord, in this sense, is any combination of notes that are played simultaneously. Pieces are generally not evolved all at once, but rather should be divided into logical excerpts and evolved separately. Evolving an entire piece is generally undesirable as it would likely create a search space too large for a GA to search effectively in a reasonable amount of time.

The parameters for the GA are those which have empirically led to convergence on the most fit individuals and were tuned manually. Our GA has a population size of 300 and uses binary tournament selection with two-point crossover. The crossover rate is set at 60% and the mutation rate (which mutates a single random chord) is set at 7%. To increase the overall performance of the program,

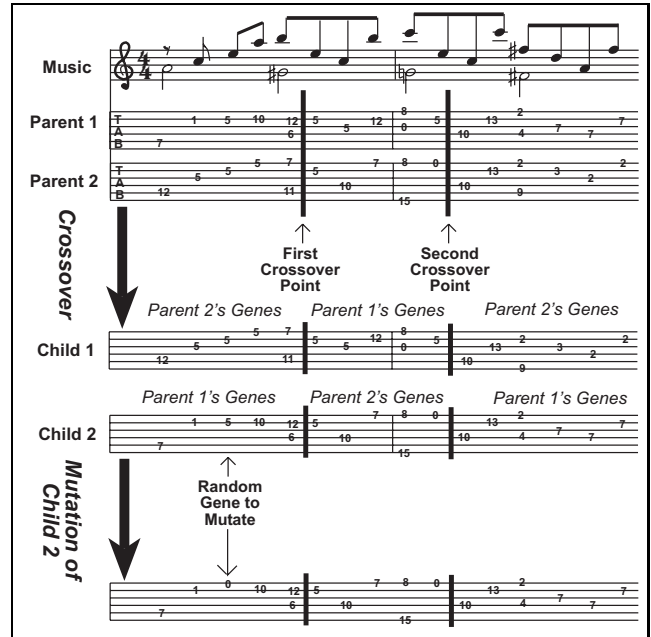


Figure 2. Obtaining children from two parents. Music from “Stairway to Heaven” by Jimmy Page and Robert Plant[1].

we found it useful to run the GA several times for 100 generations and save all of the most fit individuals found. We then run the GA once more with a population composed predominately of these individuals. Very often this final run will produce an individual more fit than any other found so far. This is often the best individual ever found as defined by our fitness function. The chance that this individual is optimal varies with the length of the excerpt and the complexity of the corresponding search space. This “seeding” scheme is often necessary because of the deceptive nature of the problem. Deception arises because each piece of music has several globally competitive but structurally divergent optima.

#### 3.3. Fitness Function

Our function analyzes a tablature in many different ways to assess its playability. This assessment is partially informed by the analysis of finger-positioning complexity of Heijink and Meulenbroek[5]. Unfortunately, it has proven very difficult to accurately capture difficulty by simply keeping track of the movement of individual fingers and our approach is therefore significantly different from that of other academic tablature generators. We acknowledge that an approach that assesses difficulty as a function of each finger’s movements is more elegant and potentially more accurate, but we found implementation of such a scheme to be impractical. We found it satisfactory, and far simpler, to devise a set of heuristics that estimate difficulty by defining general properties to which good tablature tends to adhere and measuring tablature against these. The function can be thought of as calculating two separate classes of tablature complexity, difficulty of hand/finger

movement and difficulty of hand/finger manipulation.

### 3.3.1. Hand Movement

These calculations essentially estimate the total amount of lateral hand and finger movement across the fretboard that is necessary to perform a tablature by executing simple calculations on the fret numbers. The more movement that is required, the more the function penalizes the tablature's fitness. Fitness is penalized according to three factors in this category: The total number of times that strings must be depressed, the number of frets that separate adjacent notes, and the number of frets that separate each note from the average fret of the six surrounding notes in the piece of music. Fitness is rewarded proportionally to the number of open notes, as they do not require any lateral movement by the left hand to be executed.

A number is accumulated for each one of these factors and each has an associated scaling coefficient that has been tuned so that generated tablature more accurately coincides with published tablature.

### 3.3.2. Hand Manipulation

This portion of the fitness function attempts to analyze what the left hand itself is doing and assess the difficulty of the requisite hand positions. There are two predominant methods by which a guitarist can finger multiple notes. The open chord method requires depressing strings individually for each note. Alternately, one may place a finger across a fret over several strings and use the remaining fingers for any other notes on higher frets. This is known as a barre chord.

After much manual analysis of the possible open and barre chords, we settled upon several heuristics that define valid chords of both types. At every note in a tablature, we count how many notes can be played sequentially without violating one of these heuristics. This number is then penalized according to the difficulty of the chord. The difficulty of the chord is assessed by measuring how many fingers are involved, how far they must be spread, and so on. The heuristics that define valid chords and that determine the difficulty of those chords are too numerous to enumerate here. It is also by no means true that our heuristics capture the target concept defined by the mobility of the human hand, but rather that they approximate the concept well enough for our purposes.

## 4. EXPERIMENT SETUP AND RESULTS

### 4.1. Experiment Setup

A common metric for establishing the success or failure of a tablature generator is the percentage of fretboard positions in the generated tablature that are consistent with a published tablature. Over a very large body of work this should provide a rough indication of the generator's reliability, but the metric is not perfect. While playability is the main concern for a human when creating tablature,

it is not the only concern. In cases where notes can be placed on the fretboard in multiple positions without significant differences in playability, the position chosen by a professional could seem essentially arbitrary. Because each guitar string has a slight but noticeable difference in timbre, tone quality becomes a factor for notes with more than one viable position. If differences in playability are very slight, the chosen position could even be little more than the personal preference of the tablature creator.

Consequently, our results do not lend themselves to numerical representation. Our metric for success is simply that our generator never create a tablature significantly more difficult than necessary. This is hard to accurately express statistically, since there is no entirely reliable objective measure of difficulty. If there were, it would be our fitness function. We will instead demonstrate a weakness of commercial tablature generators to which our approach is resistant. We tested several different generators including a few pieces of free software, and the commercial applications *TablEdit* and *GuitarPro 4*. The *TablEdit* software generates tablature one measure at a time, and hence tends to make mistakes in phrases that cross bar lines. The *GuitarPro* software uses an unknown algorithm that seemed to produce the most professional tablatres and was used as the primary benchmark against which we subjectively evaluated our algorithm.

### 4.2. Results and Conclusion

We ran our GA on a wide variety of both popular and classical musical literature. In all, selections from 34 pieces of music were tested. The majority of tablature coincided with published tablatres and the algorithm never produced a tablature that was significantly more difficult than the published tablature. We consider the GA a success because it reliably produces playable tablature even when departing from the published versions. Figure 4 is an example of the type of departure our generator never made. It is an excerpt from jazz standard "As Time Goes By" by Herman Hupfeld.

In this case the tablature generated by the GA is the same as the published tablature. The commercial generated tablature begins differently and as a result is probably unplayable. The first six notes are placed low on the fretboard and this results in a difficult jump from the second fret to the tenth fret between the second and third beats. The human professional and the GA, however, account for the fact that the hand will have to be high on the fretboard and place early notes in the tablature in that area as well.

The reason our approach does not make the same mistakes as the commercial generator is the implicit parallelism with which the GA processes the tablature. While the GA evaluates the first six positions of the commercially generated tablature as fit when they are by themselves, when those positions are viewed in context the tablature's fitness suffers and hence tablatres containing those positions are selected less often and eventually purged from the population.

