

# ALGORITHMIC MUSIC COMPOSITION

---

Jan Dreier

June 19, 2015

RWTH Aachen University

# HISTORY

---

## Mozart's dice game

"Anleitung zum Componieren von Walzern so viele man will  
vermitteltst zweier Würfel, ohne etwas von der Musik oder  
Composition zu verstehen"



=> 45,949,729,863,572,161 different yet similar waltzes

- 1955 Hiller, Isaacson: First computer-generated composition
- 1991 Gibson, Byrne: Musical Composition Using Genetic Algorithms And Neural Networks
- 2011 Donnelly, Sheppard: Evolving Four-Part Harmony Using Genetic Algorithms

# MUSIC COMPOSITION AS SEARCH PROBLEM

---

# MUSIC COMPOSITION AS SEARCH PROBLEM

## Goal:

- Compose music without or with minimal human guidance

## Approach:

- Search the set of all possible compositions
- Return one which sounds good

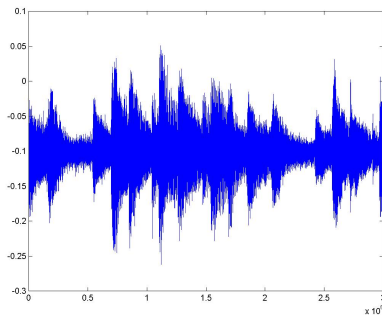
## Problems:

- How to represent music?
- Search space is very big – how to search efficiently?
- How to evaluate if something sounds good?

How can we represent music?

- Audio files
- Flat structure
- Hierarchical structure

# MUSIC REPRESENTATION



Source: <http://www.angelfire.com/art2/speech-audio-seperat/>



Love theme from  
**PRELUDE AND THE LAST HOPE**  
 IN C AND C# MINOR  
 from the Opéra Marche de Loie (*March of the Ducks*)

TRADITIONAL  
 Arranged by  
 JOHN STUMP  
 Op. 43, No. 9

Quasi Cadence  
 Slower  
 rall. *f* *p*

Reverence (Delicately) Belle  
*f* *p*

Flute Solo  
*f* *p*

Un Poco Allegro  
*f* *p*

HIGHRAIN  
*mf*

THEME  
*mf*

Finale, Act. I In 4  
*f* *pp*

Moderately Quick  
 437  
*f* *pp*

Very Slow  
*f* *pp*

Slower  
*f* *pp*

Hubato  
*f* *pp*

Surgeons only  
*mf* *f*

to the Singers  
 (legato tongue) *f* *pp*

May be sung A Capella  
 Song of Yeh Yeh *f*

Rhythmic twinkling tempo  
*f* *pp*

Shouty Ad Lib.  
*f* *pp*

Like A March  
*f* *pp*

Trumpets  
*f* *pp*

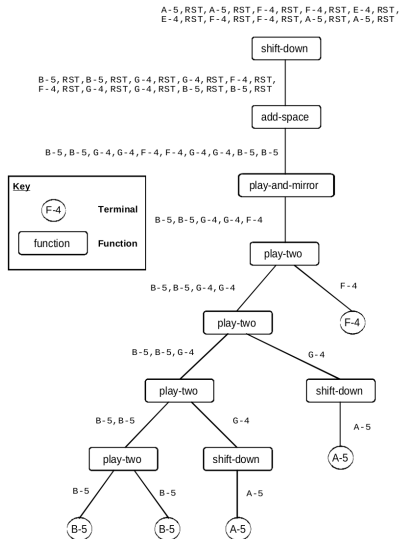
Guitar Solo  
*f* *pp*

add Arco  
 (a) legat

Violoncello and Bass

Source: <http://www.well.com/user/bryan/last.gif>

# MUSIC REPRESENTATION

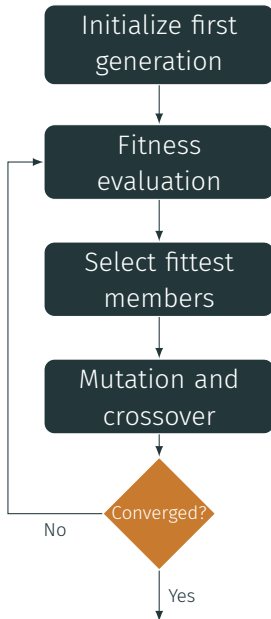


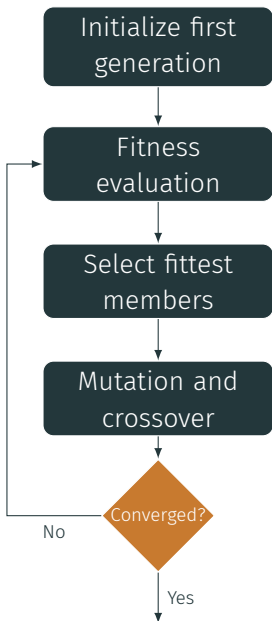
Source: <http://graphics.stanford.edu/~bjohanso/papers/gp98/johanson98gpmusic.pdf>

How can we search efficiently?

- Genetic algorithms

# GENETIC ALGORITHMS





**Example:** Maximize function  $f : \mathbb{R}^n \mapsto \mathbb{R}$

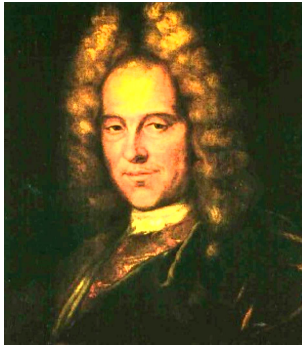
- **Initialization:** Select  $x_1, \dots, x_k \in \mathbb{R}^n$  at random
- **Fitness** of  $x$ :  $f(x)$
- **Mutation:** Shift by random  $\varepsilon \in \mathbb{R}^n$
- **Crossover** of  $x$  and  $y$ : Choose value from line-segment between  $x$  and  $y$

How can we evaluate music?

- Human based
- Rule based
- Machine learning based

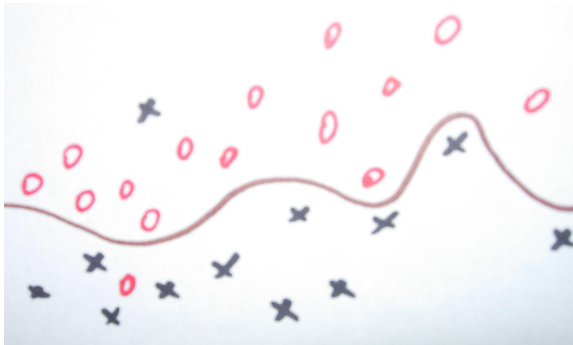
On a scale from 1 to 10, how well did you like this?

Cancel OK



Source: [https://en.wikipedia.org/wiki/Johann\\_Joseph\\_Fux](https://en.wikipedia.org/wiki/Johann_Joseph_Fux)





Source: <http://conferences.telecom-bretagne.eu/fps2012/program/slides/07.pdf>

## EVOLVING FOUR-PART HARMONY ...

---

Patrick Donnelly, John Sheppard

## **Evolving Four-Part Harmony Using Genetic Algorithms**

Applications of Evolutionary Computation, 2011, pp 273-282

### **Used techniques:**

- Flat music representation
- Genetic algorithms
- Rule based Fitness

## Music representation:

- Four parallel parts
- Each part: List of (pitch, duration) tuples
- In C-major
- Total length not fixed

## Example:



part 1: { (60, 1/4), (62, 1/4),  
(64, 1/4), (62, 1/4) }

part 2: { (48, 1/2), (52, 1/2) }

part 3: { (52, 1/2), (55, 1/2) }

part 4: { (55, 1/4), (57, 1/4),  
(59, 1/2) }

## Genetic Algorithms

**Initialize** first generation with C-major chords



**Mutation:** Apply one of these operations to a random part

Repeat Note:



Shift Random Note:



Alter Length:



**Total:** 11 operations

# CROSSOVER

Crossover: Cut and glue two elements together

The diagram illustrates the crossover technique using musical notation. It consists of three parts connected by a plus sign (+) and an equals sign (=).

**First part (left):** A four-staff musical score in common time (C). The first three staves contain a melody: the first staff has a half note C4, a quarter note D4, and a quarter note E4; the second staff has a quarter note F4, a quarter note G4, and a quarter note A4; the third staff has a quarter note B4, a quarter note C5, and a quarter note D5. The fourth staff is empty, with a common time signature 'C' and a repeat sign at the end.

**Second part (middle):** A four-staff musical score in common time (C). The first staff contains a half note C4, a quarter note D4, and a quarter note E4. The second staff contains a quarter note F4, a quarter note G4, and a quarter note A4. The third staff contains a quarter note B4, a quarter note C5, and a quarter note D5. The fourth staff is empty, with a common time signature 'C' and a repeat sign at the end.

**Third part (right):** A four-staff musical score in common time (C). The first staff contains a half note C4, a quarter note D4, and a quarter note E4. The second staff contains a quarter note F4, a quarter note G4, and a quarter note A4. The third staff contains a quarter note B4, a quarter note C5, and a quarter note D5. The fourth staff is empty, with a common time signature 'C' and a repeat sign at the end.

The plus sign (+) and equals sign (=) indicate that the first part is added to the second part to produce the third part. The third part is the result of the crossover, where the melody from the first part is combined with the melody from the second part.



Rule Based Fitness

What does good music need?

- Melody (Should be catchy)
- Harmony (Melodies should interact nicely)
- Rhythm (Emphasize meter)
- Structure (Intro, outro, reoccurring themes, ...)
- Timbre/Intonation

⇒ Music theory provides tools to enforce these constraints

- **Leap Height:** Two consecutive notes should not have an interval larger than a 9th
- **Voice Crossing:** An upper part should always play higher than a lower part
- **Opening/Closing Chord:** The piece should start and end with a C-major chord
- **Intervals:** Pure and dissonant intervals should be avoided
- **Total:** 15 rules

$$\text{fitness}(\text{rule}_i) = \frac{n_i - v_i}{n_i}$$

$$\text{total fitness} = \sum_i (\omega_i \cdot \text{fitness}(\text{rule}_i))$$

- $n_i$  = Number of places where rule  $i$  could be violated
- $v_i$  = Number of places where rule  $i$  actually is violated

## EXAMPLE

$$\text{fitness}(\text{rule}_i) = \frac{n_i - v_i}{n_i}$$

- $n_i$  = Number of places where rule  $i$  could be violated
- $v_i$  = Number of places where rule  $i$  actually is violated



### Leap Height:

- $n_i$  = number of leaps = 10
- $v_i$  = number of large leaps = 1
- $\text{fitness}(\text{Leap Height}) = \frac{9}{10}$

### Voice Crossing:

- $n_i$  = num. of parallel notes = 4
- $v_i$  = number of crossings = 2
- $\text{fitness}(\text{Voice Crossing}) = \frac{1}{2}$

### Pro:

- Strong music-theoretical foundation
- Works well in practice

### Contra:

- Humans need to define the rules for each genre (Cannot be automated)
- Some genres are hard to express by rules

We discussed:

- Genetic algorithms
- Rule based fitness functions

Using these tools it is possible to generate interesting music without human interaction.

## Papers:

- Patrick Donnelly, John Sheppard  
Evolving Four-Part Harmony Using Genetic Algorithms  
Applications of Evolutionary Computation, 2011, pp 273-282

## Images:

- [https://en.wikipedia.org/wiki/Johann\\_Joseph\\_Fux](https://en.wikipedia.org/wiki/Johann_Joseph_Fux)
- <http://conferences.telecom-bretagne.eu/fps2012/program/slides/07.pdf>
- <http://www.angelfire.com/art2/speech-audio-seperat/>
- <http://www.well.com/user/bryan/last.gif>
- <http://graphics.stanford.edu/~bjohanso/paper-s/gp98/johanson98gpmusic.pdf>
- <https://www.youtube.com/watch?v=fK2MCXpDWB4>



END