AI Hero - O Algoritmo Genético para Composição de Melodias

Matheus Bitarães de Novaes

I. Introdução

Este relatório irá descrever o desenvolvimento de um algoritmo genético para composição de melodias, como trabalho final da disciplina de Computação Evolucionaria, ministrada pelo **PPGEE - Programa de Pós Graduação da Escola de Engenharia** da **UFMG - Universidade Federal de Minas Gerais**.

A proposta deste trabalho é criar um sistema que, de acordo com parâmetros de entrada, irá gerar e executar uma melodia musical. Nas próximas sessões serão descritas as decisões tomadas para a modelagem do problema de otimização, as dificuldades encontradas e os resultados obtidos.

II. Arquitetura

A arquitetura, conforme descrito na figura 1 consiste num sistema que receberá informações com relação à execução do programa e irá gerar uma lista de notas em formado MIDI para que depois sejam executadas.

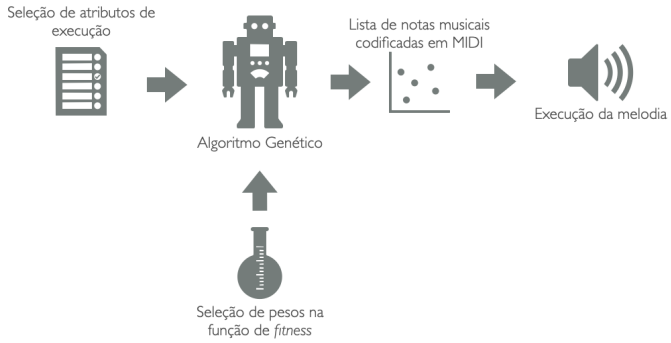

Fig. 1. Arquitetura proposta

Os valores de entrada podem se dividir em dois tipos: os atributos de execução e os pesos das funções de *fitness*. Os atributos de execução são:

- **Nota central (Tom)**: Especifica qual será a nota de referência da melodia.
- **BPM**: Número de batidas por minuto em que a melodia deverá ser executada
- **Numero de compassos**: Quantidade de compassos a serem executados
- **Escala**: Define qual será a escala utilizada como base para gerar a melodia
- **Sequencia de acordes**: Define a sequência de acordes que será tocada em cada compasso

Os atributos de *fitness* serão discutidos mais a frente, na sessão de avaliação dos indivíduos.

O algoritmo genético irá gerar uma melodia para cada compasso em formato MIDI, que é um sinal digital que transmite em tempo real informações de como um sinal de áudio deve ser excitado, bem como outros dados de estado do instrumento, de acordo com as entradas tocadas em um controlador. No formato MIDI, as notas possuem uma correspondência numérica, conforme podemos ver na figura 2

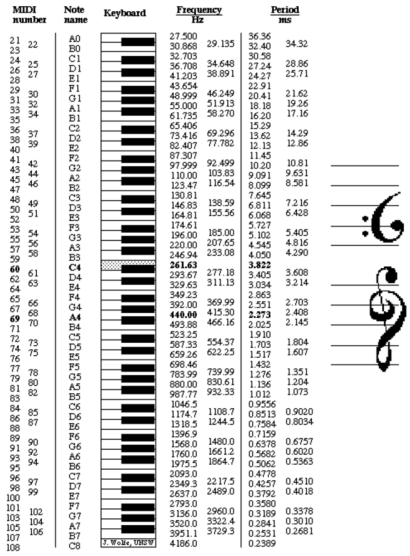

Fig. 2. Tabela de correspondência entre notas, frequência e valores MIDI

Esta notação então foi utilizada para definirmos qual nota será tocada. A biblioteca *fluidsynth* (disponível em http://www.fluidsynth.org/), em python, foi utilizada para a leitura dos sinais MIDI e sintetização de som, de acordo com as notas especificadas.

Tendo a arquitetura definida, vamos explicar em detalhes a implementação do algoritmo genético.

III. O Algoritmo Genético

A. A população

1) Representação: Cada indivíduo da população representará a melodia referente a um compasso, que é o intervalo assinalado na figura 3. Um compasso é um agrupamento de pulsos definido no começo da partitura. Cada agrupamento possui o mesmo número de pulsos [1]. Esta divisão pode ser vista na figura 3

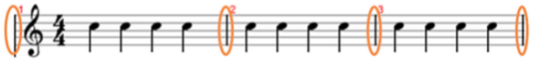

Fig. 3. Compasso: divisão de uma musica em partes de igual duração no tempo.

Cada individuo será representado por um vetor de 32 posições, referentes às notas fusas, que possuem a duração de 1/32 de um compasso, como pode ser visto na figura 4. Cada posição deste vetor poderá receber um valor que vai de -1 ate *n*, onde *n* representa o valor a ser acrescido ao valor da nota central - 12 (uma oitava abaixo). A figura 5 mostra um exemplo desta representação com a nota central em C4 (Dó).

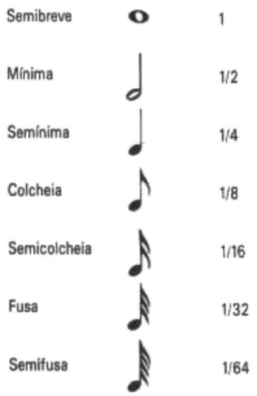

Fig. 4. Duração de notas musicais em relação a um compasso.

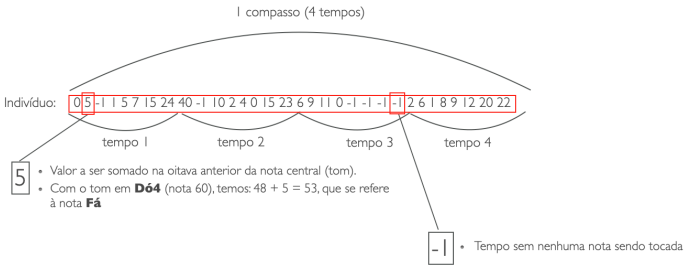

Fig. 5. Representação de um indivíduo da população

2) Inicialização: A inicialização da população poderia ser de três maneiras:

- **Totalmente Aleatória**: Indivíduos serão gerados sem nenhum direcionamento prévio. [2]
- **Semi-Aleatória**: Indivíduos serão gerados de uma forma aleatória, porém passarão por certos filtros que forçarão uma adequação prévia [2]
- **Inicialização com melodias já existentes**: Indivíduos serão gerados de acordo com um banco de melodias [2]

A forma escolhida para este trabalho foi a inicialização semi-aleatória de 100 indivíduos. A inicialização semi-aleatória acontece em dois principais passos:

1) Notas aleatórias dentro de uma escala pré definida são atribuídas às posições dos vetores.
2) É aplicado um filtro de padrões rítmicos pré definidos para garantia de coerência dos tempos das notas dentro do compasso.

Este banco de padrões rítmicos é manualmente definido e serve para pré-definir os intervalos de notas aceitos em um compasso. Isso limita que os indivíduos possuam sequencias de notas em tempos desagradáveis musicalmente.

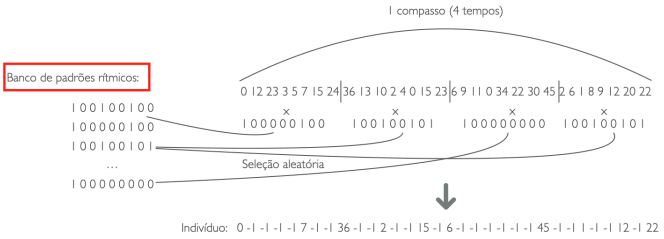

Fig. 6. Inicialização da população de acordo com o banco de padrões rítmicos

B. Seleção

A seleção é por meio de torneio. Na seleção por torneio, um grupo de indivíduos é escolhido aleatoriamente e o melhor indivíduo deste grupo é selecionado. Quando maior o grupo, maior a probabilidade de selecionar indivíduos com *fitness* acima da média [3]. No nosso caso, estamos selecionando um grupo de 30 indivíduos (30% da população) e destes, 2 pais são selecionados para o cruzamento, conforme descrito na figura 7

Fig. 7. Seleção por torneio

C. Cruzamento

O cruzamento é com base no *crossover* de um ponto [3], porém com a particularidade de que os pontos de corte podem ser apenas entre tempos (baseado na implementação descrita em [2]), conforme demonstrado na figura 8

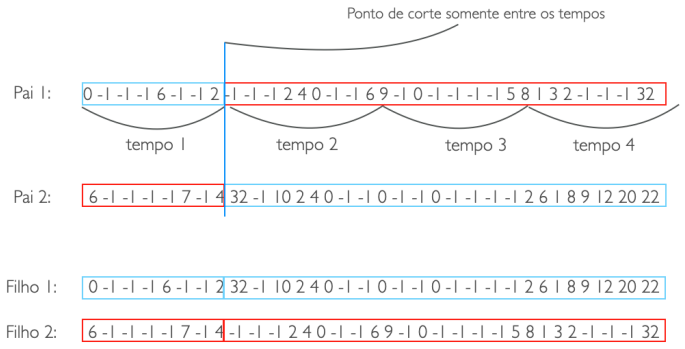

Fig. 8. Cruzamento por *crossover* de um ponto com pontos definidos

Foi definida uma probabilidade de cruzamento de 50%.

D. mutação

Foram consideradas duas opções para a mutação. A primeira, uma mutação construtiva [2]. Tal mutação iria gerar uma nota de ligação no individuo mutado, colocando uma nota intermediária entre duas notas da melodia. Esta mutação mostrou-se ineficiente e foi descartada pois gerava melodias bastante imprevisíveis. A segunda opção aplicada foi a mutação por troca de notas. Cada nota do individuo a ser mutado teria uma probabilidade de 2% de ser trocada por uma outra nota 4 semi-tons menor ou 4 semi-tons maior, conforme descrito na figura 9. Cada individuo teria 5% de chance de mutação. Esta segunda opção apresentou melhores resultados e portanto foi a escolhida.

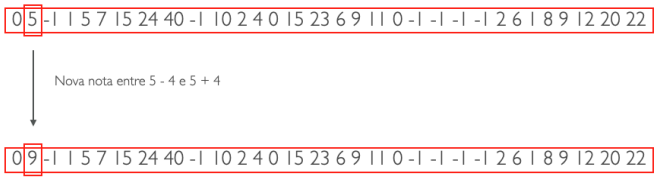

Fig. 9. Mutação por mudança de notas

E. Avaliação dos indivíduos

Foram consideradas três opções para avaliação da função de *fitness*:

- **Sem Aptidão**: Seria uma AG sem uma avaliação de *fitness* e portanto, nenhuma pressão seletiva. Para isso, é necessário garantir que todos os indivíduos inicializados na população seriam melodias que já possuem um "*fitness* mínimo".[2]
- **Interativo**: Seria um modelo onde cada melodia seria valiada por um agende humano. [2]
- **Automático**: Avaliação por meio de heurísticas para determinar se a melodia está de acordo com regras conhecidas da musica.[2]

A opção escolhida foi a de um *fitness* automático, calculado através de heurísticas. Foram definidas então 8 funções objetivo para a avaliação de uma melodia, conforme pode ser descrito na figura 10. Todas elas possuem valores que variam de 0 a 1.

Função	Definição	Saída
f1	Porcentagem de notas coincidentes com notas do acorde	0..1
f2	Melodia começa com uma nota do acorde	0..1
f3	Porcentagem de intervalos no compasso	0..1
f4	Número de notas iguais seguidas maior do que 2	0 ou 1
f5	Notas próximas a uma nota de referencia	0..1
f6	Variedade de notas	0..1
f7	Notas em sequencia	0..1
f8	Notas nos tempos ou contra tempos	0..1

Fig. 10. Lista de funções objetivo

1) Porcentagem de notas coincidentes com notas do acorde: Avalia a porcentagem de notas na melodia que estão coincidentes com as notas do acorde definido para aquele compasso.

2) Melodia começa com uma nota do acorde: Avalia se o indivíduo em questão possui sua primeira nota coincidente com a primeira nota do acorde definido para aquele compasso.

3) Porcentagem de intervalos no compasso: Retorna a porcentagem de notas vazias (com valores iguais a -1) em um indivíduo.

4) Número de notas iguais seguidas maior do que 2: Avalia se há 3 ou mais notas iguais consecutivas. Caso sim, retorna 1, caso não, retorna 0.

5) Notas próximas a uma nota de referencia: Avalia se as notas da melodia estão próximas a uma nota de referência. Esta nota de referencia é informada pelo usuário na hora da definição dos pesos da função objetivo. O peso desta função está fixado com o valor 50.

6) Variedade de notas: Avalia a porcentagem de notas únicas na melodia.

7) Notas em sequência: Avalia a porcentagem de notas em sequencia, crescente ou decrescente.

8) Notas nos tempos ou contratempos: avalia a porcentagem das notas que estão sendo tocadas em cima dos tempos ou na subdivisão deste intervalo, os contratempos.

Considerando estas 8 funções objetivo, podemos entender este problema então como uma otimização multi-objetivo. Porém, para este trabalho, decidiu-se por transformá-la em uma otimização mono objetivo, criando apenas uma função objetivo como a soma ponderada das oito funções:

$$f(x) = w_1 f_1(x) + w_2 f_2(x) + w_3 f_3(x) + w_4 f_4(x) + w_5 f_2 5(x) + w_6 f_6(x) + w_7 f_7(x) + w_8 f_8(x)$$

Os pesos w são definidos através de uma interface gráfica, *a priori*.

F. Critério de Parada

O critério de parada escolhido foi pelo numero máximo de gerações que, neste trabalho, foi de 150 gerações.

G. Limitações da modelagem

A modelagem escolhida apresenta as seguintes limitações:

- Todas as notas possuem a mesma velocidade (intensidade sonora). Não é possível criar uma nota com um volume mais alto que a outra.
- Todas as notas possuem a mesma duração, que é a duração da nota fusa. Não é possível gerar uma nota que se estende por um tempo inteiro, por exemplo
- Pode ser tocada apenas uma nota por vez. Não é possível que se execute uma melodia com mais de uma nota sendo tocada ao mesmo tempo.

IV. A Interface gráfica

Foi desenvolvida uma interface gráfica em python, utilizando a biblioteca open source *Kivy* (disponível em https://kivy.org/#home). Esta interface gráfica permite que o usuário insira os parâmetros de entrada do algoritmo bem como os pesos das funções de *fitness*.

Fig. 11. Interface gráfica do AI Hero

V. Análise dos resultados

Conforme pode ser visto no Video de demonstração (url: https://youtu.be/8aPmBeKnWrA), se iniciarmos a execução com peso 0 para todas os atributos de fitness, temos um resultado praticamente aleatório, uma vez que não há nenhuma pressão seletiva. A melodia irá ser composta de notas aleatórias dentro da escala escolhida para a execução. Porém, conforme vamos dando peso a alguns objetivos, vemos que o algoritmo consegue convergir para resultados perceptivelmente mais coerentes com a decisão tomada pelo usuário, o que indica um sucesso na execução do algoritmo genético. Ao final do Video de demonstração temos uma execução de 12 compassos, na escala de blues maior. É um resultado coerente com as funções objetivo estipuladas, porém ainda não é um resultado totalmente satisfatório musicalmente.

VI. Conclusão

Através deste trabalho foi possível acompanhar o desenvolvimento de um algoritmo genético para geração de melodias. A principal dificuldade ocorre na forma de se avaliar os indivíduos da otimização, uma vez que não existe um ponto ótimo para a arte e, portanto, não há uma função objetivo a ser avaliada. Foi tentando portanto aproximar esta função inexistente por heurísticas

É possível perceber uma certa coesão entre as notas e que as melodias tendem a alçar os objetivos estipulados pelas funções de *fitness*. Porém, ainda há bastante espaço para melhorias. Uma delas pode ser a adição de novas funções de *fitness* que tentem privilegiar frases melódicas conhecidas, tendendo assim para uma sequencia de notas mais agradável, em média.

Para maiores informações, o código fonte do projeto se encontra neste repositório do github

References

[1] P. Nickol, *Learning To Read Music 3rd Edition: How to make sense of those mysterious symbols and bring music alive*. Hachette UK, 2008.

[2] A. R. R. d. Freitas, "Música evolutiva: Uma abordagem computacional para composição algorıtmica.," 2011.

[3] A. M. Andrew, *Introduction to evolutionary computing, by ae eiben and je smith (natural computing series), springer, berlin, 2003, hardback, xv+ 299 pp., isbn 3-540-40184-9 (£ 30.00); book review; book review*, 2004.