UNIVERSIDADE FEDERAL DO PARANÁ

DÉBORAH BAPTISTA PILATO

APLICAÇÃO DE HEURÍSTICAS E METAHEURÍSTICAS NA COMPOSIÇÃO DE SÉRIES DODECAFÔNICAS

CURITIBA

2017

DÉBORAH BAPTISTA PILATO

APLICAÇÃO DE HEURÍSTICAS E METAHEURÍSTICAS NA COMPOSIÇÃO DE SÉRIES DODECAFÔNICAS

Dissertação apresentada como requisito parcial à obtenção do grau de Mestre em Ciências, no Curso de Pós-Graduação em Métodos Numéricos em Engenharia – Programação Matemática, Setores de Tecnologia e de Ciências Exatas, da Universidade Federal do Paraná.

Orientador: Prof. Dr. Paulo Henrique Siqueira

Coorientador: Prof. Dr. José Eduardo Pécora Jr.

CURITIBA

2017

DADOS INTERNACIONAIS DE CATALOGAÇÃO NA PUBLICAÇÃO (CIP)
UNIVERSIDADE FEDERAL DO PARANÁ
SISTEMA DE BIBLIOTECAS – BIBLIOTECA CIÊNCIA E TECNOLOGIA

Pilato, Déborah Baptista
Aplicação de heurísticas e metaheurísticas na composição de séries dodecafônicas. / Déborah Baptista Pilato. – Curitiba, 2017.
1 recurso on-line : PDF.

Dissertação (Mestrado) - Universidade Federal do Paraná, Setores de Tecnologia e de Ciências Exatas, Programa de Pós-Graduação em Métodos Numéricos em Engenharia.
Orientador: Prof. Dr. Paulo Henrique Siqueira. Coorientador: Prof. Dr. José Eduardo Pécora Jr.

1. Heurística. 2. FORTRAN (Linguagem de programação de computador). I. Siqueira, Paulo Henrique. II. Pécora Jr., José Eduardo. III. Universidade Federal do Paraná. Programa de Pós-Graduação em Métodos Numéricos em Engenharia. IV. Título.

Bibliotecária: Roseny Rivelini Morciani CRB-9/1585

MINISTÉRIO DA EDUCAÇÃO
SETOR CIÊNCIAS EXATAS
UNIVERSIDADE FEDERAL DO PARANÁ
PRÓ-REITORIA DE PESQUISA E PÓS-GRADUAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO MÉTODOS NUMÉRICOS EM ENGENHARIA

ATA Nº

ATA DE SESSÃO PÚBLICA DE DEFESA DE MESTRADO PARA A OBTENÇÃO DO GRAU DE MESTRE EM MÉTODOS NUMÉRICOS EM ENGENHARIA

No dia vinte e tres de Novembro de dois mil e dezessete às 14:00 horas, na sala Laboratório de Computação - LDC, Centro de Estudos em Engenharia Civil - CESEC/UFPR, campus Centro Politécnico, foram instalados os trabalhos de arguição da mestranda **DEBORAH BAPTISTA PILATO** para a Defesa Pública de sua dissertação intitulada **APLICAÇÃO DE HEURÍSTICAS E METAHEURÍSTICAS NA COMPOSIÇÃO DE SÉRIES DODECAFÔNICAS**. A Banca Examinadora, designada pelo Colegiado do Programa de Pós-Graduação em MÉTODOS NUMÉRICOS EM ENGENHARIA da Universidade Federal do Paraná, foi constituída pelos seguintes Membros: PAULO HENRIQUE SIQUEIRA (UFPR), JOSÉ ROBERTO FREGA (UFPR), LUZIA VIDAL DE SOUZA (UFPR). Dando início à sessão, a presidência passou a palavra a discente, para que a mesma expusesse seu trabalho aos presentes. Em seguida, a presidência passou a palavra a cada um dos Examinadores, para suas respectivas arguições. A aluna respondeu a cada um dos arguidores. A presidência retomou a palavra para suas considerações finais. A Banca Examinadora, então, reuniu-se e, após a discussão de suas avaliações, decidiu-se pela APROVAÇÃO da aluna. A mestranda foi convidada a ingressar novamente na sala, bem como os demais assistentes, após o que a presidência fez a leitura do Parecer da Banca Examinadora. A aprovação no rito de defesa deverá ser homologada pelo Colegiado do programa, mediante o atendimento de todas as indicações e correções solicitadas pela banca dentro dos prazos regimentais do programa. A outorga do título de mestre está condicionada ao atendimento de todos os requisitos e prazos determinados no regimento do Programa de Pós-Graduação. Nada mais havendo a tratar a presidência deu por encerrada a sessão, da qual eu, PAULO HENRIQUE SIQUEIRA, lavrei a presente ata, que vai assinada por mim e pelos membros da Comissão Examinadora.

Curitiba, 23 de Novembro de 2017.

PAULO HENRIQUE SIQUEIRA
Presidente da Banca Examinadora (UFPR)

JOSÉ ROBERTO FREGA
Avaliador Externo (UFPR)

LUZIA VIDAL DE SOUZA
Avaliador Interno (UFPR)

UNIVERSIDADE FEDERAL DO PARANÁ
SETOR DE TECNOLOGIA
Programa de Pós-Graduação Métodos Numéricos em Engenharia

MINISTÉRIO DA EDUCAÇÃO
SETOR CIÊNCIAS EXATAS
UNIVERSIDADE FEDERAL DO PARANÁ
PRÓ-REITORIA DE PESQUISA E PÓS-GRADUAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO MÉTODOS NUMÉRICOS EM ENGENHARIA

TERMO DE APROVAÇÃO

Os membros da Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em MÉTODOS NUMÉRICOS EM ENGENHARIA da Universidade Federal do Paraná foram convocados para realizar a arguição da dissertação de Mestrado de **DEBORAH BAPTISTA PILATO** intitulada: **APLICAÇÃO DE HEURÍSTICAS E METAHEURÍSTICAS NA COMPOSIÇÃO DE SÉRIES DODECAFÔNICAS**, após terem inquirido a aluna e realizado a avaliação do trabalho, são de parecer pela sua APROVAÇÃO no rito de defesa.

A outorga do título de mestre está sujeita à homologação pelo colegiado, ao atendimento de todas as indicações e correções solicitadas pela banca e ao pleno atendimento das demandas regimentais do Programa de Pós-Graduação.

Curitiba, 23 de Novembro de 2017.

PAULO HENRIQUE SIQUEIRA
Presidente da Banca Examinadora (UFPR)

JOSÉ ROBERTO FREGA
Avaliador Externo (UFPR)

LUZIA VIDAL DE SOUZA
Avaliador Interno (UFPR)

MINISTÉRIO DA EDUCAÇÃO
SETOR CIÊNCIAS EXATAS
UNIVERSIDADE FEDERAL DO PARANÁ
PRÓ-REITORIA DE PESQUISA E PÓS-GRADUAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO MÉTODOS NUMÉRICOS EM ENGENHARIA

LISTA DE PRESENÇA

Mestrando(a): DEBORAH BAPTISTA PILATO

Dissertação intitulada: APLICAÇÃO DE HEURÍSTICAS E METAHEURÍSTICAS NA COMPOSIÇÃO DE SÉRIES DODECAFÔNICAS

Data: 23/11/2017

Hora: 14:00

Local: Centro de Estudos em Engenharia Civil - CESEC/UFPR, campus Centro Politécnico

Sala: Laboratório de Computação - LDC

Banca constituída pelos seguintes professores:
PAULO HENRIQUE SIQUEIRA
JOSÉ ROBERTO FREGA
LUZIA VIDAL DE SOUZA

NOME LEGÍVEL	ASSINATURA
José Eduardo Pécora Jr	
Luzia Vidal de Souza	
José Roberto Frega	
LUIZ ANTONIO R. DE SANTANA	
Inara D. M. de Abreu	Ingrid Abreu.
Jéssica Soares Alves	

LILIANA MADALENA GRAMANI
Coordenação do Programa de Pós Graduação em MÉTODOS NUMÉRICOS EM ENGENHARIA

A Deus.

AGRADECIMENTOS

Agradeço primeiramente a Deus, fonte de toda sabedoria e força, que me permitiu chegar até aqui.

À minha família, em especial à minha mãe e minha avó, que me deram o suporte necessário em cada passo.

Ao professor Paulo Henrique pela orientação, apoio, compreensão e valiosas dicas que enriqueceram este trabalho.

Ao professor José Eduardo Pécora, por ter contribuído com seus conhecimentos e sua boa vontade na realização deste trabalho.

Ao professor Liduíno Pitombeira, sempre disposto a compartilhar comigo seu amplo conhecimento.

Ao professor Rodrigo Marques, pelo incentivo e por ter me ensinado o pouco que sei sobre o universo da teoria musical.

Ao Secretário Jair dos Anjos, pelas infindáveis ajudas e pelo interesse na prestatividade sempre presente.

E finalmente aos amigos e demais pessoas que de uma forma ou de outra me ajudaram a chegar aqui.

Em memória de meu pai, falecido nesse ano.

RESUMO

O objetivo deste trabalho é desenvolver, a partir do uso de heurísticas e metaheurísticas, meios de elaborar e obter compassos musicais que satisfaçam as caraterísticas do método de composição dodecafônico. Para isso, a série de notas utilizada é a cromática, optando-se por gerar compassos 2 por 4, 3 por 4, 5 por 4 e 2 por 2 a partir de mínimas, semínimas, colcheias e semicolheias. A linguagem de programação escolhida para desenvolver os algoritmos é a linguagem Fortran, e o software MuseScore foi utilizado para representar as soluções obtidas na linguagem de partitura. Para que a aplicação das metaheurísticas fosse possível no contexto musical, os problemas aqui expostos foram reformulados de modo a tornarem-se equivalentes aos problemas característicos dessa outra área, como o Problema do Caixeiro Viajante e o Problema da Mochila, associando as notas musicais a objetos ou a cidades, permitindo que a resolução seguisse de maneira semelhante ao o que ocorre em tais problemas. Os resutados obtidos foram satisfatórios, pois os algoritmos retornaram compassos de acordo com o estipulado pelas restrições iniciais, mostrando que é possível compor as séries e compassos a partir de métodos heurísticos e metaheurísticos. Com as soluções obtidas em mãos, o objetivo é o de que estas sirvam de suporte ao compositor, que terá total liberdade de fazer alterações, descartar soluções que não lhe interessem e trabalhar com novos valores, buscando assim outras alternativas. Ou seja, os algoritmos desenvolvidos têm como principal objetivo servir de ferramenta no processo composicional, preservando sempre a criatividade e a liberdade presentes na criação.

ABSTRACT

The objective of this work is to develop, from the use of heuristics and metaheuristics, means of elaborating and obtaining musical measures that satisfy the characteristics of the dodecaphonic composition method. For this, the series of notes used is the chromatic one, being chosen to generate measures 2 by 4, 3 by 4, 5 by 4 and 2 by 2 from half notes, quarter notes, eighth notes and sixteenth notes. The programming language chosen to develop the algorithms is the Fortran language, and the MuseScore software was used to represent the solutions obtained in the score language. In order for the application of metaheuristics to be possible in the musical context, the problems presented here were reformulated so as to become equivalent to the problems characteristic of this other area, such as the Traveling Salesman Problem and the Backpack Problem, associating musical notes with objects or cities, allowing resolution to follow in a manner similar to what occurs in such problems. The obtained results were satisfactory, because the algorithms returned bars according to the stipulations of the initial restrictions, showing that it is possible to compose the series and measures from heuristic and metaheuristic methods. With the solutions obtained at hand, the objective is that these support the composer, who will have complete freedom to make changes, discard solutions that do not interest him and work with new values, thus seeking other alternatives. That is, the algorithms developed have as main objective to serve as a tool in the compositional process, always preserving the creativity and freedom present in creation.

LISTA DE FIGURAS

Figura 1: Passo a passo *3-OPT* 20
Figura 2: Estrutura básica de um algoritmo 22
Figura 3: AG com um ponto de cruzamento 23
Figura 4: AG com dois pontos de cruzamento 23
Figura 5: Mutação em dois pontos 24
Figura 6: Exemplo do PVC 27
Figura 7: Série Cromática 31
Figura 8: Posição de cada nota de partitura 31
Figura 9: Figuras Rítmicas 32
Figura 10: Proporção entre as figuras rítmicas 33
Figura 11: Compasso 3 por 4 34
Figura 12: Compasso 2 por 2 34
Figura 13: Compasso 4 por 4 34
Figura 14: Quadrado mágico de Babbitt 36
Figura 15: Transformações Isomórficas 37
Figura 16: Relação entre notas e valores 39
Figura 17: Passo 1 - Algoritmo 1 41
Figura 18: Aplicação do método *2-OPT* 42
Figura 19: Critério de parada - Algoritmo 1 43
Figura 20: Solução do algoritmo 1 44
Figura 21: Indivíduo inicial X 45
Figura 22: Formando o primeiro compasso 46
Figura 23: Completando os demais compassos 47
Figura 24: Cruzamento (Crossover) 48
Figura 25: Seleção dos melhores adaptados 49
Figura 26: Perda de diversidade 50
Figura 27: Mutação - Algoritmo 2 51
Figura 28: Solução do algoritmo 2 51
Figura 29: Compassos formados pelo algortimo 2 52
Figura 30: Solução obtida pelo algoritmo 2 na partitura 52
Figura 31: Primeira solução do algoritmo 3 53
Figura 32: Segunda solução do algoritmo 3 54
Figura 33: Terceira solução do algoritmo 3 54
Figura 34: Valores das figuras rítmicas no algoritmo 4 55

Figura 35: População Inicial – Algoritmo 4 55
Figura 36: Controle da perda de diversidade 56
Figura 37: Mutação - Algoritmo 4 57
Figura 38: Mutação - Verificação do primeiro compasso...59
Figura 39: Restrição sobre o último termo...60
Figura 40: Restrição sobre o último termo do terceiro compasso 61
Figura 41: Solução obtida através do algoritmo 5 62
Figura 42: Valores das figuras rítmicas - Algoritmo 6..62
Figura 43: População Inicial – Algoritmo 6 63
Figura 44: Mutação - Algoritmo 6 65
Figura 45: Solução ótima obtida através do algoritmo 6..66

LISTA DE ABREVIATURAS E SIGLAS

PCV	- Problema do Caixeiro Viajante
AG	- Algoritmos Genéticos
PM	- Problema da Mochila
PMC	- Problema da Mochila Compartimentada

Sumário

1 INTRODUÇÃO 15

1.1 JUSTIFICATIVA... 15

1.2 OBJETIVOS... 16

1.3 LIMITAÇÕES DO TRABALHO 17

1.4 ESTRUTURA DO TRABALHO 17

2 REVISÃO BIBLIOGRÁFICA 19

2.1 HEURÍSTICAS E METAHEURÍSTICAS 19

2.2 K-OPT... 19

2.3 ALGORITMOS GENÉTICOS... 20

2.4 PROBLEMA DO CAIXEIRO VIAJANTE 25

2.5 PROBLEMA DA MOCHILA... 27

2.5.1 PROBLEMA DA MOCHILA COMPARTIMENTADA 28

3 NOÇÕES BÁSICAS DE COMPOSIÇÃO MUSICAL 30

3.1 NOTAS MUSICAIS 30

3.2 FIGURAS RÍTMICAS... 32

3.3 FÓRMULAS DE COMPASSO 33

3.4 MÉTODOS DE COMPOSIÇÃO 35

4 APRESENTAÇÃO DOS RESULTADOS 39

4.1 ALGORITMO 1 39

4.2 ALGORITMO 2 44

4.3 ALGORITMO 3 52

4.4 ALGORITMO 4 54

4.5 ALGORITMO 5 58

4.6 ALGORITMO 6 62

5 CONSIDERAÇÕES FINAIS 67

5.1 RECOMENDAÇÕES PARA TRABALHOS FUTUROS... 67

REFERÊNCIAS... 69

1 INTRODUÇÃO

Em um mundo tão complexo e rico, que permite constantemente o contato com novos conhecimentos e aprimoramentos, a inovação acaba sendo uma consequência lógica. Isso porque, sendo uma característica nata ao ser humano, a curiosidade leva-o a lugares antes nunca visitados, visto que o caminho gerado por perguntas e respostas bem estruturadas conduzem a novos horizontes, fortalecendo muitas vezes o poder de criação.

No meio empresarial, a produção e a competitividade caminham lado a lado, de modo que o desejo de liderar o mercado no qual atuam é o principal objetivo das companhias. Nesse sentido, inúmeros problemas surgem, sejam eles associados à produção, à logística, à distribuição e a inúmeros outros, sempre buscando um lucro maior ou então um menor custo.

Frequentemente, na modelagem de novos problemas e na busca por soluções encontram-se diversos obstáculos. Sejam estes devidos às restrições que possam existir, ou então pela dificuldade de combinar tais restrições com as variáveis existentes na formulação do problema. De uma forma ou de outra, o objetivo principal é sempre o mesmo: obter o melhor resultado possível. Nesse sentido, surge o conceito de otimização, que de acordo com Soares (1997, p.4) é “um mecanismo de análise de decisões complexas, envolvendo seleção de valores para variáveis (...)” cuja “intenção é encontrar a melhor solução, respeitando, se necessário, restrições de viabilidade (...)”.

Na busca por tal solução surgem as Heurísticas e Metaheurísticas, onde as últimas, por exemplo, são inspiradas na física, biologia, ciências sociais, entre outros, e partem de métodos ou técnicas aproximativas que não garantem a solução ótima, mas chegam a soluções satisfatórias para determinados problemas.

Neste trabalho são propostas algumas aplicações de tais procedimentos na composição musical, baseando-se em condições pré-estipuladas, e buscando assim desenvolver uma ferramenta que permita ao compositor opções de sequências de notas e tempos de duração para as tais, que se adaptem às condições pré-definidas por ele.

1.1 JUSTIFICATIVA

Com as diversas opções de aplicação que os métodos metaheurísticos permitem, a justificativa geral deste trabalho consiste em apresentar uma aplicação no campo da música, realizando associações entre estes problemas de naturezas distintas, e no entanto, mostrando que é possível resolvê-los de maneira semelhante, utilizando os mesmos meios. Sem deixar, no entanto, de respeitar as características particulares de cada um.

1.2 OBJETIVOS

Partindo do sistema dodecafônico, o objetivo inicial deste trabalho é o de gerar sequências de notas partindo da escala cromática, de maneira que sejam respeitadas distâncias pré-estabelecidas entre duas notas consecutivas. Esta distância citada não é invariável, ficando a cargo do compositor e se adequando ao seu objetivo final. Além disso, a partir sequência gerada deseja-se criar compassos que as agrupem, formando assim pequenas séries musicais.

Para que essa sequência seja gerada e que os compassos sejam criados, esses problemas precisam ser modelados de modo a se adequarem aos métodos heurísticos e metaheurísticos. Dessa forma, as notas passaram a integrar um vetor solução, de modo que a cada uma delas foi associado um valor numérico. Essa formulação equivale ao Problema do Caixeiro Viajante, permitindo a aplicação do método heurístico em sua resolução. A relação que surge então é a de que cada nota musical representa uma cidade, e o objetivo é o de visitá-las, passando uma única vez por cada uma. Da mesma forma ocorre para a criação dos compassos, que levando em consideração as figuras rítmicas associadas a cada nota – que definem seu tempo de duração –, foi escrita de forma vetorial a solução, de modo que cada uma de suas entradas representa a duração associada a cada nota. Deseja-se então preencher os compassos com tais valores, criando assim uma relação desse problema com o Problema da Mochila, cujo objetivo é o de encher uma mochila com objetos, de modo que seja respeitada sua capacidade e que o valor (ou a relevância) dos objetos levados seja a maior possível.

Dessa forma, o objetivo é do modelar os problemas e desenvolver algoritmos que trabalhem na busca da solução ótima, retornando séries e compassos musicais que se enquadrem nas condições desejadas.

1.3 LIMITAÇÕES DO TRABALHO

A principal limitação encontrada no decorrer deste trabalho diz respeito à função responsável por gerar valores aleatórios que compõe a solução inicial no software *Fortran*. O que ocorre é que os valores não são de fato aleatórios, e sim pseudoaleatórios, de modo que ao seguirem distribuição uniforme ocorre a repetição de sequências, o que limita o campo de busca da solução ótima, uma vez que parte sempre do mesmo vetor inicial.

Na tentativa de contornar esse problema foi utilizada para criação da população inicial valores gerados no software *Excel* e apenas lidos pelo software *Fortran*. No entanto, isso não se fez de maneira automática, sendo necessário criar os aleatórios desejados externamente e inseri-los em um arquivo para a posterior leitura. Tal ferramenta resolveu o problema da repetição da população inicial, mas se mostrou insuficiente do ponto de vista prático.

Outra tentativa nesse mesmo sentido foi a de gerar valores aleatórios a partir de dados fornecidos pelo próprio usuário, como minutos e segundos atuais. Tal ferramenta se mostrou eficaz para os valores que definem as posições de aplicação das heurísticas e metaheurísticas, mas não apresentou bons resultados para a geração da população inicial, por se tratar de vetores com 12 entradas, dificultando assim a diversidade entre eles.

1.4 ESTRUTURA DO TRABALHO

O presente trabalho está organizado em cinco capítulos, incluindo esta introdução.

No segundo capítulo é apresentado um apanhado de pesquisas e definições a respeito das heurísticas e metaheurísticas que foram utilizadas no decorrer do projeto.

O capítulo 3 é dedicado a alguns temas na área da música, buscando introduzir (de maneira simplificada) os conceitos mais básicos que são necessários para a compreensão dos resultados que aparecem na sequência. Além disso, esse capítulo aborda alguns sistemas de composição, dando ênfase ao dodecafonismo.

Já o capítulo 4 apresenta a estrutura dos algoritmos que foram desenvolvidos, bem como os objetivos de cada um e as definições passo a passo com os respectivos resultados obtidos.

E, finalmente, o capítulo 5 – baseado em tudo o que foi mostrado –, apresenta considerações finais acerca do exposto, e faz recomendações para trabalhos futuros, que porventura venham a dar continuidade ao que foi aqui iniciado.

2 REVISÃO BIBLIOGRÁFICA

2.1 HEURÍSTICAS E METAHEURÍSTICAS

As metaheurísticas são processos que visam a melhoria de uma solução, tendendo a solução ótima, e que consistem na aplicação de uma heurística subordinada em cada fase, a qual necessita ser adequada a cada problema em particular. Segundo Chaves (2003), a metaheurística tem como primordial característica a capacidade de se evadir de ótimos locais dando flexibilidade às restrições da função objeto. As heurísticas fundamentam-se no aperfeiçoamento do movimento exercido, procurando um ótimo local, porém sempre oferecem a mesma solução se originadas de um mesmo ponto de partida, sendo assim limitadas. Pesquisas elaboradas durante décadas a cerca do desempenho de métodos heurísticos resultaram na formulação de estratégias genéricas para sua formulação, e tais estratégias foram denominadas de metaheurísticas. Dessa forma, as metaheurísticas visam suprir as limitações encontradas na aplicação de procedimentos heurísticos, aprimorando soluções, deixando um ótimo local com intuito de buscar um ótimo global, mesmo que isto acarrete temporariamente na perda do valor da função objetivo.

De maneira geral, uma metaheurística pode ser vista como um conjunto “de algoritmos que podem ser aplicados a diferentes problemas de otimização com relativamente poucas modificações para torná-los adaptados a um problema específico." (METAHEURISTICS NETWORK, 2011).

O presente trabalho utiliza a heurística *k-opt* na resolução do Problema do Caixeiro Viajante (PCV) e a metaheurística Algoritmos Genéticos (AGs) na resolução do Problema da Mochila (PM), os quais são definidos a seguir.

2.2 K-OPT

Conforme proposto por Lin e Kernighan (1973), o algoritmo *k-opt* visa a melhoria de uma solução a partir da troca de combinações dentro de sua própria vizinhança. Para isso, partindo de uma solução - obtida anteriormente a partir de um outro método – o método remove dela k arcos, e os substitui por k outros. O procedimento é uma generalização do método *2-opt*, que apaga duas arestas de uma solução, transformando-a em dois sub percursos. Religam-se então os pontos formando uma nova solução, que substitui a anterior caso apresente melhores resultados.

Quanto maior for o número k, maior também será a eficiência do algoritmo. Logo, em um problema com n pontos, se o algoritmo *n-opt* fosse aplicado o ótimo certamente seria obtido, pois todas as trocas possíveis seriam testadas. No entanto, para um valor alto de n, também alto tornar-se-á o tempo computacional despendido. Por essa razão é que na literatura costuma-se encontrar aplicações que utilizam os métodos 2 e 3-opt, e a última delas está exemplificada na Figura 1.

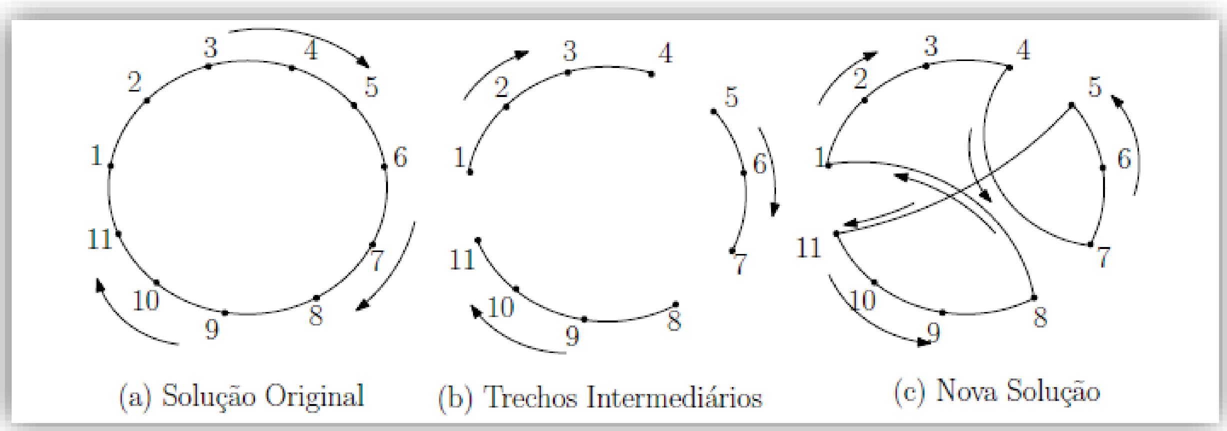

Figura 1: Passo a passo *3-OPT*
Fonte: VITOR, 2015)

O algoritmo aqui desenvolvido utiliza o método *2-opt*, uma vez que a vizinhança do problema é relativamente pequena e tal método se faz suficiente.

2.3 ALGORITMOS GENÉTICOS

Concebidos em 1960 por John Holland, os AGs surgiram com o objetivo inicial de estudar os fenômenos relacionados à adaptação das espécies e também desenvolver uma maneira de relacioná-los com os computadores (MITCHELL, 1997). Já nos anos 50 e 60 diversos biólogos trabalhavam no desenvolvimento de simulações computacionais de sistemas genéticos, e o ponto inicial dos AGs foi em 1975, quando Holland publicou o livro “*Adaptation in Natural and Artificial Systems*”.

Na década de 80, David E. Goldberg obteve o primeiro sucesso em aplicação industrial com AGs.

Os algoritmos genéticos possuem uma larga aplicação em muitas áreas científicas, entre as quais podem ser citados problemas de otimização de soluções, aprendizado de máquinas, desenvolvimento de estratégias e fórmulas matemáticas, análise de modelos econômicos, problemas de engenharia, diversas aplicações na biologia como simulação de bactérias, sistemas imunológicos, ecossistemas, descoberta de formato e propriedades de moléculas orgânicas (MITCHELL 1997 *apud.* Pozo *et. al.*).

Inspirados na Teoria da Evolução das Espécies de Charles Darwin, os AGs assemelham-se aos processos naturais de seleção, presentes em todos os grupos de seres vivos que ocupam nosso planeta. Tal teoria afirma que buscando a sobrevivência os indivíduos de uma mesma população competem entre si, e os mais aptos terão mais descendentes, propagando assim seus genes.

Para a aplicação desse algoritmo fazem-se necessárias algumas condições iniciais, que segundo Pozo (2011) são as seguintes:

- Representações das possíveis soluções do problema no formato de um código genético;
- População inicial que contenha diversidade suficiente para permitir ao algoritmo combinar características e produzir novas soluções;
- Existência de um método para medir a qualidade de uma solução potencial;
- Um procedimento de combinação de soluções para gerar novos indivíduos na população;
- Um critério de escolha das soluções que permanecerão na população ou que serão retiradas desta; e,
- Um procedimento para introduzir periodicamente alterações em algumas soluções da população. Desse modo mantém-se a diversidade da população e a possibilidade de se produzir soluções inovadoras para serem avaliadas pelo critério de seleção dos mais aptos.

Uma solução tem uma população inicial de indivíduos – geralmente gerados de maneira aleatória – que são chamados de “pais”, e sobre ela aplicam-se os operadores genéticos, criando assim combinações, as quais são chamadas de “filhos”, e a Figura 2 mostra o fluxograma desse algoritmo.

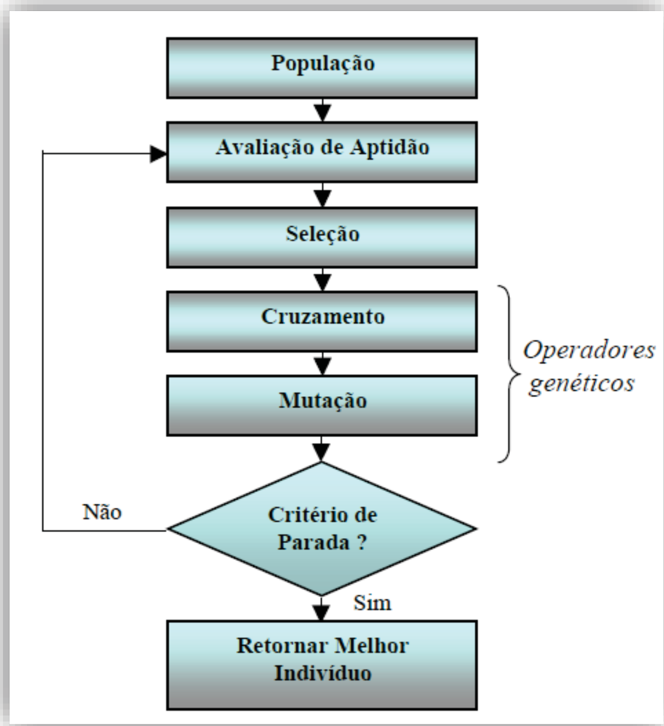

Figura 2: Estrutura básica de um algoritmo
(Fonte: POZO, A. *et al.* 2011)

Mesmo nem sempre encontrando o ótimo global, os AGs na maioria das vezes obtêm uma solução próxima a solução ótima para o problema, o que se torna aceitável visto que sua implementação despende tempo computacional inferior se comparado a outros métodos existentes.

Os operadores genéticos citados anteriormente são três: reprodução, cruzamento (*crossover*) e mutação, e nos problemas aqui discutidos são utilizados os dois últimos. O cruzamento utiliza dois indíviduos pais e combina seus materiais genéticos, gerando assim novos indivíduos, que herdam suas caraterísticas genéticas. Esta mistura é feita tentando imitar (em um alto nível de abstração) a reprodução de genes em células. Para efetuar o cruzamento entre os indivíduos são definidos aleatoriamente as posições, sendo geralmente utilizados 1 ou 2 pontos. As Figuras 3 e 4 apresentam exemplos desse procedimento.

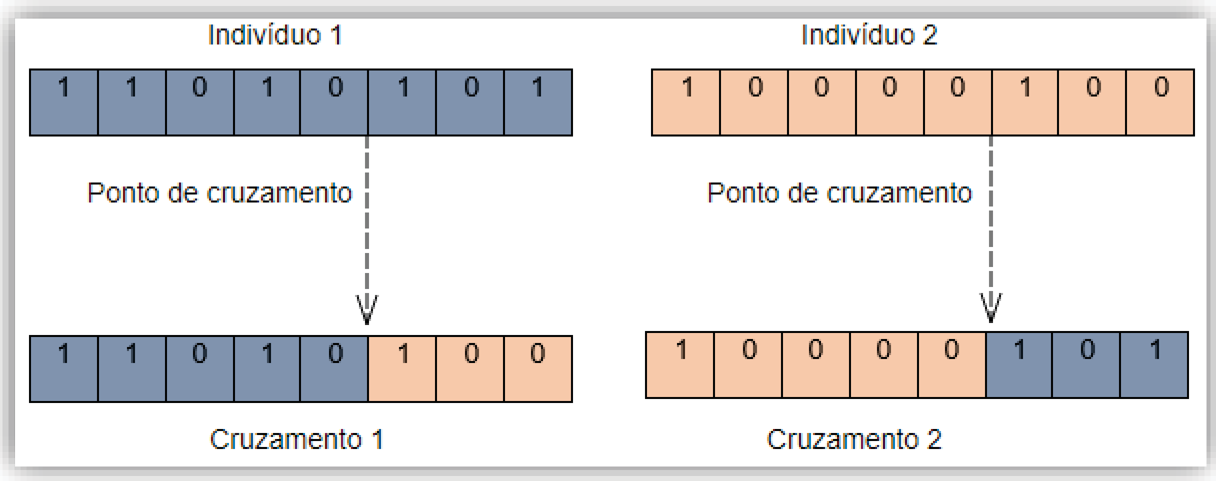

Figura 3: AG com um ponto de cruzamento

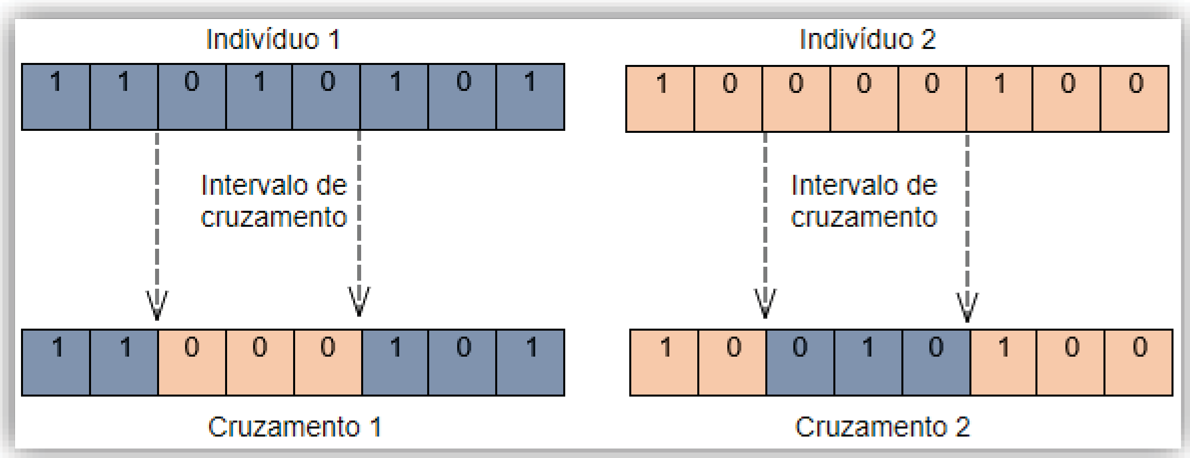

Figura 4: AG com dois pontos de cruzamento

Note que de acordo com o que mostra a Figura 3, o método de cruzamento em um único ponto define a posição que irá mesclar as características dos pais. Sendo então definida aleatoriamente a posição 6 (por exemplo), o que ocorre é que o cruzamento 1 herda até a sexta posição os genes do indivíduo 1, e a partir dessa posição genes do indivíduo 2. Com o cruzamento 2 ocorre o inverso, de modo que até a sexta posição ele é composto pelos genes do indivíduo 2 e nas demais posições por genes do indivíduo 1.

Já a Figura 4 exemplifica o cruzamento em dois pontos do vetor solução, de modo que o intervalo compreendido entre tais posições é mesclado entre os genes dos pais, ficando o cruzamento 1 com genes do indivíduo 1 até a posição 3 e após a posição 6, e com genes do indivíduo 2 nesse intervalo. De maneira inversa, o cruzamento 2 herda os genes do indivíduo 2 até a posição 3 e após a posição 6, e no intervalo entre as duas posições possui os genes do indivíduo 1.

Como é natural observar, em populações pequenas há grandes chances de perda de diversidade, o que gera um impedimento à melhora da solução. Já em populações muito grandes, há um alto custo computacional. Outro problema também associado à perda de diversidade é a ocorrência de incesto, e uma saída para isso é evitar o cruzamento entre indivíduos muito semelhantes.

Outra forma de contornar esse problema é a aplicação da mutação, que modifica aleatoriamente alguma característica do indivíduo sobre o qual é aplicada. Isso é importante porque ela ajuda a manter diversidade da população, garantindo que qualquer ponto do espaço de busca pode ser alcançado. Além disso, pode ser uma importante ferramenta no caso de estagnação na busca pela solução ótima, escapando de pontos de máximos locais.

Esse método pode ser aplicado em um ou mais genes do indivíduo, ficando a critério do autor, e a Figura 5 exemplifica a mutação em 2 pontos.

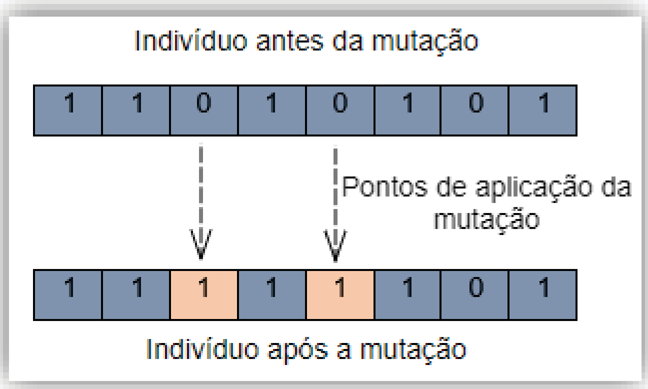

Figura 5: Mutação em dois pontos

O que ocorre é que são definidos ao acaso duas posições de troca, e a mutação age alterando o valor dessas posições. Sendo um vetor composto de modo binário, a entrada irá valer zero ou um. Aplicada a mutação, e sendo x_i e x_j os pontos sobre os quais será aplicado o operador genético, ocorre o que é mostrado na equação 1, sendo que x_{k^*} é o ponto anterior e x_k posterior à mutação.

$$x_{k*} = \begin{cases} 0, se\ x_k = 1 \\ 1, se\ x_k = 0 \end{cases} \quad \forall k = i, j \tag{1}$$

Logo, o operador age invertendo o valor analisado. Se ele é igual a 0 passa a valer 1, e se é 1 passa a valer 0.

Uma análise dos resultados deve ser feita após a aplicação dos operadores genéticos, concluindo se os indivíduos gerados por eles apresentam ou não uma melhora na busca por uma solução ótima. A *função fitness* (ou função objetivo) calcula o valor de aptidão de cada indivídiuo, medindo quão próxima uma solução está do valor desejado. Por isso é essencial que esta função seja muito representativa e diferencie na proporção correta as más soluções das boas, para evitar que se fique inutilmente à procura do ótimo global.

Quanto ao critério de parada do algoritmo, este pode ser definido de diversas formas. Como listado por Siqueira (2016, p.102) podem ser eles: "o alcance de um número máximo de gerações; solução ótima encontrada; perda de diversidade; ou convergência: nas últimas *k* gerações não houve melhora na aptidão.".

Já com relação à representação dos elementos, ela geralmente "se resume à utilização de cadeias (*strings*) de comprimento *l*, formadas por caracteres de um determinado alfabeto. O caso mais comum é o binário, onde o alfabeto é composto pelos símbolos 1 e 0." (BENEVIDES, 2011).

Por meio de todos esses passos, o método vai realizando a melhora gradual da solução obtida a partir da evolução e combinação entre indivíduos.

2.4 PROBLEMA DO CAIXEIRO VIAJANTE

De acordo com Zamboni (1997), a primeira vez que o termo "Problema do Caixeiro Viajante" foi utilizado nos meios matemáticos foi entre 1931 e 1932. Mas apenas em 1954 teria sido aplicado em um problema de larga escala por Dantzig, Fulkerson e Johnson.

Clássico entre os problemas combinatórios, o Problema da Caixeiro Viajante (PCV) consiste em encontrar o roteiro de menor distância ou custo que passa por um conjunto de cidades, sendo cada cidade visitada apenas uma vez, e finalizando na cidade de origem.

Como citam Cunha, Bonasser e Abrahão (2002), o PCV pertence à categoria conhecida como *NP-hard*, o que significa que possui ordem polinomial não determinístico, uma vez que não existem algoritmos determinísticos polinomiais capazes de resolvê-lo. Logo, apenas os problemas de pequeno porte podem ser solucionados de forma ótima. Problemas de grande porte tornam-se inviáveis se forem utilizados métodos exatos, em virtude do esforço computacional que seria exigido para resolvê-los (BENEVIDES, 2001). Devido a inviabilidade de métodos exatos, os métodos heurísticos tornaram-se o principal meio de resolução desse tipo de problema, servindo de maneira particular a um problema.

Para definir o problema do caixeiro viajante deve-se considerar um Grafo G (N; E) em que N= {1; 2; ...; n} é o conjunto de nós ou vértices, A é o conjunto de arcos $C = [c_{ij}]$ é uma matriz tal que c_{ij} representa o custo associado a aresta que liga os vértices *i* e *j*. A matriz $X = [x_{ij}]$ é composta pelas variáveis de decisão do problema, e é formada como mostra a Equação 2.

$$x_{ij} = \begin{cases} 1, se\ o\ arco\ a_{ij}\ \in rota \\ 0, se\ o\ arco\ a_{ij}\ \notin rota \end{cases} \tag{2}$$

Desta forma, a formulação de Programação Linear Inteira para o problema, devido a Golden et al., de 1977, (BODIN et al., 1983 *apud* BENEVIDES, 2001), pode ser escrita como mostram as Equações de 3 a 7.

Minimizar

$$\sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij} \tag{3}$$

Sujeito a

$$\sum_{i=1}^{n} x_{ij} = 1 \qquad (j = 1,2, \dots, n) \tag{4}$$

$$\sum_{j=1}^{n} x_{ij} = 1 \qquad (i = 1,2, \dots, n) \tag{5}$$

$$X = (x_{ij}) \in S \tag{6}$$

$$x_{ij} = 0\ ou\ x_{ij} = 1 \quad (i, j = 1,2, \dots, n) \tag{7}$$

As equações 2 e 3 garantem que exatamente um arco (i, j) tem origem no nó *i* da rota e exatamente um arco (i, j) é direcionado para cada arco *j* da rota. Já o conjunto *S* presente na penúltima restrição impede a formação de sub-rotas, a qual é chamada de restrição de sub-rotas, e pode ser representado pela equação 8, na qual R representa todo conjunto não vazio de {2, 3, ..., n}.

$$S = \left\{ (x_{ij} : \sum_{i \in R} \sum_{j \in R} x_{ij} \leq |R| - 1 \right\} \quad (8)$$

A Figura 6 exemplifica o PVC aplicado em um problema com 6 nós, e como foi visto, pode-se chamar os nós de cidades e as arestas de rotas, e dessa forma o objetivo seria ir de uma cidade à outra, de modo que o somatório de rotas (que é a distância total percorrida) seja o menor possível.

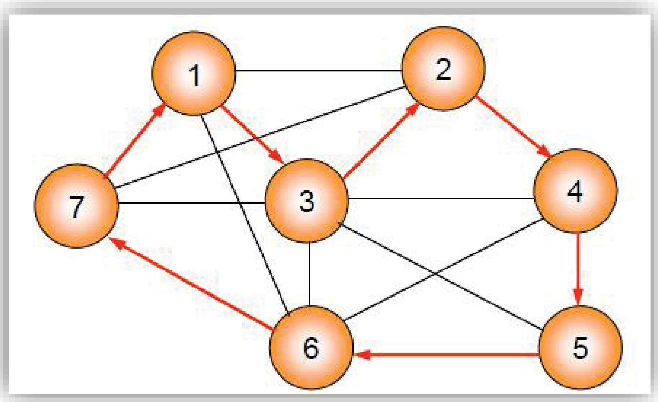

Figura 6: Exemplo do PVC
(Fonte: BENEVIDES, 2011)

2.5 PROBLEMA DA MOCHILA

Imagine que você vai fazer uma viagem e para isso deverá levar alguns itens em sua mochila. No entanto, a mochila tem uma capacidade e pode ocorrer de não caber nela tudo o que você gostaria de levar. O mais natural então é pensar que terá que renunciar a algumas em detrimento de outras, correto? A pergunta que surge agora é “o que devo levar?”

O Problema da Mochila (PM) surge justamente para buscar uma resposta para a pergunta anterior, ou seja, deseja definir quais itens serão levados sem que a capacidade da mochila seja excedida. Para isso, no entanto, também é necessário que seja atribuído para cada item um valor de importância, porque afinal de contas há objetos fundamentais e outros não.

Suponha então que a capacidade da mochila é W e que temos n itens distintos. Seja x_1, x_2, ..., x_n a quantidade de itens que está sendo carregada, w_1, w_2, ..., w_n os pesos e v_1, v_2, ..., v_n os valores dos respectivos objetos.

Deseja-se então maximizar a função representada pela Equação 9.

$$\sum_{i=1}^{n} v_i \,.\, x_i \tag{9}$$

Sujeita a restrição presente na Equação 10.

$$\sum_{i=1}^{n} w_i \,.\, x_i \leq W, onde\ x_i\ \in \mathbb{N} \tag{10}$$

Garante-se assim a busca pela solução que acumule o maior valor dentre os objetos que a mochila irá comportar, respeitando a limitação quanto ao peso desses objetos. Portanto, o PM é um problema de programação linear inteira e, assim como o PVC, é classificado como NP-hard devido a característica de não poder ser dada por uma função polinomial a quantidade de passos para sua resolução.

2.5.1 PROBLEMA DA MOCHILA COMPARTIMENTADA

Para que o Problema da Mochila Compartimentada (PMC) possa ser compreendido pode-se partir da ideia do PM acrescentando um detalhe. Imagine que dentre os itens que deseja levar há classes distintas entre eles, como por exemplo comidas, roupas, livros, entre outros, e que você deseja alocá-los em bolsos distintos dentro da mochila.

Um custo c_k é então associado a cada compartimento caso ele seja usado com algum item da classe k e o somatório de todos os compartimentos não pode exceder a capacidade total da mochila. O PMC consiste em determinar as capacidades adequadas de cada compartimento e como estes devem ser carregados de modo que o valor de utilidade total seja máximo (CARVALHO, 2015).

Nota-se então que o PMC é uma variação do clássico problema da mochila e pode ser enunciado da seguinte forma: “Imagine que você possui uma mochila que suporta uma capacidade máxima de peso e gostaria de preenchê-la com vários objetos. No entanto, os objetos possuem pesos e valores distintos entre si”. E continua com a seguinte restrição: “Além disso, a mochila possui diversos compartimentos, com capacidades limitadas e distintas entre si. A inclusão de um novo compartimento implica na perda da capacidade total da mochila, e possui um custo fixo associado" (MARQUES e ARENALES, 2002).

Nos próximos capítulos são abordados conceitos musicais, as ideias básicas que estão ligadas à composição das séries e compassos que se deseja gerar e a conexão entre isso e as formulações do PCV e PM expostas, bem como os algoritmos desenvolvidos a partir dos métodos heurísticos e metaheurísticos e os resultados obtidos.

3 NOÇÕES BÁSICAS DE COMPOSIÇÃO MUSICAL

Falar sobre composição musical é o mesmo que falar sobre criar, inventar, reunir ferramentas, conhecimentos, experiências e sentimentos para transmitir algo novo. Para Koelreutter (1984) essa produção é chamada de “Arte Experimental”, pois não é possível prever os resultados e impactos que isso causará no ouvinte. Isso porque a individualidade de quem ouve também é um fator importante nesse processo, de modo que suas vivências, lembranças e história influenciarão no resultado daquilo que chega aos seus ouvidos. Música é afeto, é sentimento, é sensação. “O processo de criação é uma experiência emocional associada aos sentidos.” (HINDEMITH, 1969 *apud* SANTOS, 2008).

A composição musical é estruturada conforme a manipulação de elementos harmônicos, melódicos e estilísticos, sendo cada um deles de relevância no desenvolvimento dessa construção. O objetivo aqui não é o de detalhar todos os elementos envolvidos nesse processo, mas sim o de abordar temas triviais, que fazem parte da compreensão do que vem a seguir. Os elementos fundamentais (notas, compassos, figuras rítmicas, entre outos) e alguns métodos composicionais são abordados com o objetivo de garantir o entendimento do que foi desenvolvido nesse trabalho.

3.1 NOTAS MUSICAIS

Notas musicais podem ser definidas como os elementos mínimos de um som. Quando uma corda vibra e as moléculas de ar ao seu redor se movimentam, essa movimentação ocorre na mesma vibração na corda. O ouvido humano então capta tal vibração, processando-a e atribuindo a ela um som (MÚSICA, 2017). O que ocorre então é que cada som diferente atribuído pelo cérebro representa cada uma das notas musicais.

Representando a “vibração do ar”, cada nota tem uma duração, uma frequência associada (medida em hertz). Pode-se dizer que as notas servem então para nomear tais frequências, e se não existissem, o que consta nas partituras seriam números imensos para representar esses valores.

Chama-se de escala cromática a escala de 12 sons formada pelas 7 notas padrão da escala diatônica (Dó, Ré, Mi, Fá, Sol, Lá, Si) acrescidas dos 5 tons intermediários (com bemol e sustenido), formando então a sequência representada na Figura 7.

Dó – Dó# – Ré – Ré# – Mi – Fá – Fá# – Sol – Sol# – Lá – Lá# – Si

Figura 7: Série Cromática

Tal sequência teve origem no século XI sendo proposta pelo monge italiano Guido d'Arezzo, que associou as primeiras sílabas do Hino a São João Batista às notas que iniciavam cada verso, formando os seis primeiros sons da escala maior diatônica. As sílabas Ut, Ré, Mi, Fá, Sol e Lá passaram a servir como referenciais para os sons a serem cantados (FREIRE, 2005). Mais tarde a sílaba Ut foi substituída por Dó e a sílaba Si foi acrescentada, formando a escala de notas utilizada hoje.

Nos países anglo-saxônicos as notas são representadas pelas sete primeiras letras do alfabeto, de modo que a sequência A – B – C – D – E – F – G equivale as notas: Lá – Si – Dó – Ré – Mi – Fá – Sol. Os sinais # e ♭ são utilizados para representar o sustenido e o bemol, que marcam as alterações cromáticas das notas. Já nos países de língua germânica, além das sete letras utiliza-se também o H, que representa a nota Si, e o B representa a nota Si bemol. Já para o sustenido e o bemol usam-se as terminações *is* e *es.*

Sendo a escrita musical uma importante ferramenta nas comunicações entre músicos, muitas formas de representá-la surgiram. Além das já citadas, uma de notável importância é a partitura, que após muitos anos de mudanças e evoluções chegou ao que temos hoje, onde para cada nota musical a posição específica que a representa pode ser visualizada na Figura 8.

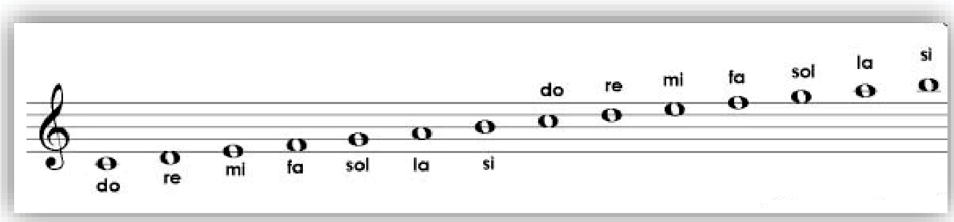

Figura 8: Posição de cada nota de partitura

No entanto, a Figura 8 demonstra apenas qual a nota deve ser tocada em cada caso, não definindo a duração dessa nota, e para isso é que surgem os símbolos que representam tais durações, e estes são apresentados a seguir.

3.2 FIGURAS RÍTMICAS

São as figuras rítmicas que indicam o tempo de duração de uma nota musical, e as mais usadas são a semibreve, a mínima, a semínima, a colcheia, a semicolcheia, a fusa e a semifusa, cujos símbolos que as representam podem ser visualizados na Figura 9. Há também uma relação de proporcionalidade entre elas, indicando que uma figura sempre vale metade da anterior. Ou seja, duas semínimas equivalem a uma mínima, por exemplo, e essa relação pode ser observada na Figura 10. Assim sendo, dentro dos compassos e dependendo da fórmula envolvida, cada figura rítmica assumirá um tempo de duração, que marcado na partitura sobre a posição da nota desejada representará o tempo que ela irá soar.

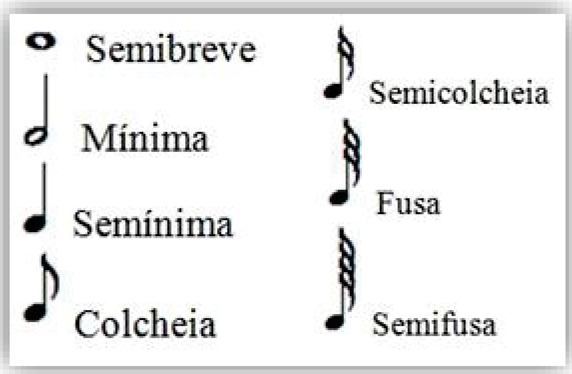

Figura 9: Figuras Rítmicas

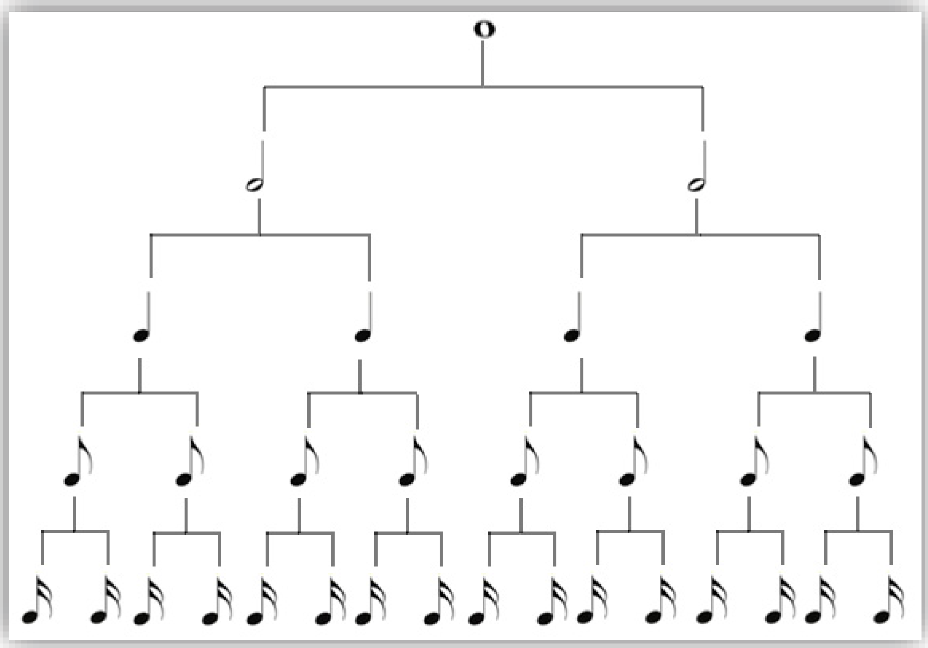

Figura 10: Proporção entre as figuras rítmicas

Veja que o equivalente a uma semibreve podem ser duas mínimas, quatro semínimas, oito colcheias, e assim por diante. É justamente essa relação que define qual figura rítmica vale uma unidade de tempo em uma determinada partitura (definida pela fórmula de compasso), e a partir dela obtêm-se os valores de todas as demais. Logo, em um compasso no qual a figura rítmica que equivale a uma unidade de tempo é a semínima, a colcheia equivale a meia unidade, e dessa forma soa metade do tempo que a primeira. Para que fique mais clara essa compreensão a seguir é apresentada a explicação sobre as fórmulas de compasso.

3.3 FÓRMULAS DE COMPASSO

A figura rítmica que representa uma unidade de tempo é indicada logo no início da partitura, após o símbolo da clave. As oito figuras mais comuns foram vistas, e a cada uma delas associa-se o que é chamado de “fórmula de compasso”.

A fórmula de compasso é representada por dois números, um sobre o outro, como se formassem uma fração. O de cima (numerador) representa quantos tempos o compasso terá, e o de baixo (denominador) diz qual a figura rítmica que equivale a uma unidade. Assim sendo, o denominador “1” estará ligado a semibreve, “2” a mínima, “4” a semínima e assim por diante. Fato que se explica pela relação de proporcionalidade existente entre as notas. Dessa forma, um compasso 3 por 4 indica que nele haverá 3 tempos e que a semínima vale 1 tempo, e na partitura ele aparece como mostra a Figura 11.

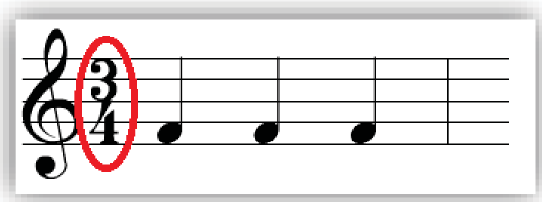

Figura 11: Compasso 3 por 4

De maneira geral os compassos aparecem indicados dessa forma, mas isso nem sempre ocorre, podendo em alguns momentos não ser representado por uma fração, e sim por um símbolo. Tais casos são exceções, e ocorrem quando o compasso em questão é o 2 por 2 – que pode ser abreviado por um C cortado – e o 4 por 4 – que pode ser abreviado por um C –, e ambos os casos são representados nas Figuras 12 e 13.

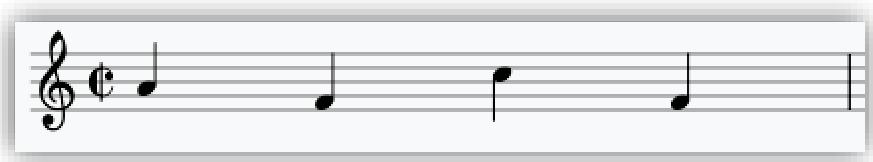

Figura 12: Compasso 2 por 2

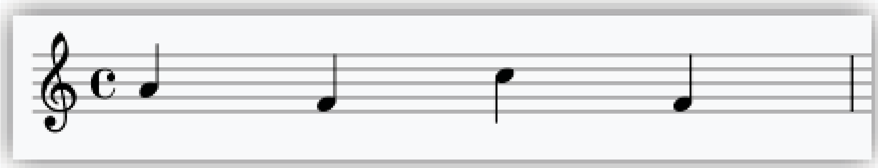

Figura 13: Compasso 4 por 4

Note que a Figura 11 apresenta não só a fórmula de compasso, mas também uma possibilidade de o completar. A figura rítmica utilizada é a semínima, que devido ao denominador 4 equivale a uma unidade de tempo, e que ao se repetir por três vezes na posição Fá preenche o espaço determinado pelo numerador, e indica que essa nota deve soar três vezes consecutivas, de modo que cada uma dure uma unidade de tempo.

3.4 MÉTODOS DE COMPOSIÇÃO

No instante em que o tema composição musical é abordado, abre-se um vasto horizonte, trazendo diversos meios e formas, bem como diversos estilos e estruturas. Com o objetivo de buscar uma sintetização dessas formas, para Corrêa (2004), pode-se considerar dois tipos de processos composicionais: aqueles possuidores de um centro de atração tonal e os que prescindem de sua existência. Aqui, é dado ênfase ao método de composição atonal, mais especificamente ao dodecafonismo.

Até meados do século XIX predominavam as composições tonais, que pressupõem uma hierarquia entre as notas musicais, de modo que há um polo de atração e as notas comportam-se seguindo essa ordem estabelecida, com agrupamentos formados a partir da sobreposição de terças. Já no final do século, passou-se a observar um enfraquecimento dos vínculos com o polo atrativo, de modo que nem mesmo o retorno da tônica no encerramento da obra se fazia necessário. Essa mudança aos poucos deu origem a sistemas diferenciados, surgindo o que foi denominado sistema atonal.

Antes disso, todavia, outras denominações foram atribuídas, e primeiramente surgiu o tonalismo livre que, para Dallin (1975), representa o procedimento composicional em que apenas a tônica exerce poder sobre as demais notas, as outras onze possuem a mesma ordem de hierarquia. Já o atonalismo livre nega qualquer forma de dependência entre as notas, carecendo, no entanto, de definições mais profundas e teoria bem fixada, já que a liberdade composicional empregada acaba causando a perda de relações entre a construção como um todo.

Nesse sentido surgiu o dodecafonismo, que visa fornecer uma estrutura ao sistema atonal. Isso porque, devido à indepêndencia contida no processo (além da carência teórica) o método estava apresentando resquícios tonais, que é justamente o oposto objetivo inicial. Foi então que Shoenberg, um dos maiores nomes da época, desenvolveu a técnica de composição intitulada *doze notas* ou *dodecafonismo*, que tem como ideia principal de que nenhuma nota pode ser repetida até que as onze demais tenham sido utilizadas.

Esse procedimento deu uma sensação de automatismo ao método, sendo criticado por limitar o processo de criação. No entanto, essa limitação é bem menos restrita do que os críticos afirmavam, uma vez que não havendo alteração quando transformada isomorficamente, as séries formadas podem ser utilizadas em qualquer de suas transposições ou em qualquer de suas transformações invariantes, isto é, na inversão, retrogradação ou retrogradação da inversão. Essas diferentes representações foram organizadas em forma de matriz, que ficou conhecida como quadrado mágico de Babbitt (KOSTKA, 1990, p. 209, apud. SOUZA, 2009). A Figura 14 apresenta essa matriz aplicada a série dodecafônica **F#, F, D, E, Eb, C, A, C#, G#, B, Bb, G**.

	I-6	**I-5**	**I-2**	**I-4**	**I-3**	**I-0**	**I-9**	**I-1**	**I-8**	**I-11**	**I-10**	**I-7**	
O-6	**F♯**	**F**	**D**	**E**	**E♭**	**C**	**A**	**C♯**	**G♯**	**B**	**B♭**	**G**	**R-6**
O-7	G	F♯	D♯	F	E	C♯	B♭	D	**A**	C	B	G♯	**R-7**
O-10	B♭	A	F♯	G♯	G	E	C♯	F	**C**	E♭	D	B	**R10**
O-8	A♭	G	E	F♯	F	D	B	E♭	**B♭**	C♯	C	A	**R-8**
O-9	A	G♯	F	G	F♯	D♯	C	E	**B**	D	C♯	B♭	**R-9**
O-0	C	B	G♯	B♭	A	F♯	D♯	G	**D**	F	E	C♯	**R-0**
O-3	E♭	D	B	C♯	C	A	F♯	B♭	**F**	G♯	G	E	**R-3**
O-11	B	B♭	G	A	G♯	F	D	F♯	**C♯**	E	D♯	C	**R11**
O-4	E	E♭	C	D	C♯	B♭	G	B	**F♯**	A	G♯	F	**R-4**
O-1	C♯	C	A	B	B♭	G	E	G♯	**D♯**	F♯	F	D	**R-1**
O-2	D	C♯	A♯	C	B	G♯	F	A	**E**	G	F♯	D♯	**R-2**
O-5	F	E	C♯	D♯	D	B	G♯	C	**G**	B♭	A	F♯	**R-5**
	RI-6	**RI-5**	**RI-2**	**RI-4**	**RI-3**	**RI-0**	**RI-9**	**RI-1**	**RI-8**	**RI.11**	**RI.10**	**RI-7**	

Figura 14: Quadrado mágico de Babbitt
(FONTE: KOTSKA, 1990 *apud.* SOUZA 2009)

Note que a linha O-6, lida da esquerda para a direita, representa a série original. Já a série retrógrada é representada por R-6, a invertida por I-6 e a retrógrada da invertida por RI-6. As demais linhas e colunas representam todas as possíveis combinações dessa série, que pode ser iniciada a partir de qualquer nota da sequência original, o que mostra que o processo não é tão limitado quanto se afirmava, sendo possível gerar inúmeras possibilidades a partir da sequência principal.

É importante relembrar, como cita SOUZA (2009), que todas as séries dodecafônicas compartilham da escala cromática, e assim sendo, a série não define as notas que serão usadas – já que todas serão –, mas sim a ordem entre tais notas. O autor ainda faz uma comparação com o DNA das células humanas, que diferem entre si não pelos componentes químicos que as formam, mas pela ordem entre eles.

A série original apresentada na Figura 14 faz parte da obra dodecafônica “Wie bin ich froh!” do op.25, de Anton Webern, e a Figura 15 apresenta as linhas e colunas O-6, R-6, I-6 e RI-6 na forma de partitura.

Figura 15: Transformações Isomórficas
(FONTE: KOTSKA, 1990 *apud.* SOUZA 2009)

Nessa composição, Webern utilizou apenas as quatro formas apresentadas, sendo que o uso de outras transformações seria totalmente possível, aumentando assim o grau de complexidade da obra.

Assim sendo, o dodecafonismo trouxe uma estrutura que conferiu firmeza ao tipo de estrutura não tonal, induzindo, por outro lado, a ausência de significados harmônicos internos. Finalizando, para Shoenberg, a utilização da música atonal abriu as portas para a livre utilização da dissonância, que para ele é a essência desse processo. E ainda Krenek (1940) considerou o dodecafonismo como “um sistema não-tonal ou atonal que exerce um controle muito rígido sobre a construção melódica e contrapontística que é compensada pela liberdade que concede ao campo harmônico”.

Uma vez apresentados os conceitos musicais necessários, segue-se com a apresentação dos problemas formulados e seus objetivos, a relação estabelecida entre esses problemas e os métodos heurísticos e metaheurísticos, bem como o desenvolvimento dos algoritmos necessários para resolvê-los e as respectivas soluções obtidas, representadas em formato tanto vetorial quanto musical, para que seja possível visualizar a integração entre todos os tópicos até aqui abordados.

4 APRESENTAÇÃO DOS RESULTADOS

Em todos os algoritmos detalhados a seguir, o conjunto de indivíduos é a escala cromática, e obviamente não é possível utilizar de seus próprios nomes na linguagem de programação. Assim sendo, para cada uma das notas é associado um número decimal, de forma que a primeira nota corresponderá ao 0, a segunda ao 0,5, a terceira ao 1,0, e assim por diante, até que finalmente a última será equivalente ao 5,5. A distância entre os valores, que varia de 0,5 em 0,5, baseia-se no fato de que as notas distam meio tom de uma para outra. Logo, teremos a correspondência ilustrada na Figura 16.

C	C#	D	D#	E	F	F#	G	G#	A	A#	B
0.0	0.5	1.0	1.5	2.0	2.5	3.0	3.5	4.0	4.5	5.0	5.5

Figura 16: Relação entre notas e valores

Vale lembrar que a ordenação das notas é a única coisa que caracteriza uma série dodecafônica, o que equivale a dizer que a característica essencial de uma série é a sucessão dos intervalos entre as notas que a compõem, não as notas por si. Assim sendo, o objetivo inicial é o de gerar uma sequência de notas que utilize as doze citadas, de maneira que duas notas consecutivas estejam no mínimo a 1 tom de distância uma da outra e no máximo 3 tons de distância. Tal restrição é totalmente subjetiva e fica a critério do compositor, não existindo nenhuma razão especial para essa escolha. Na sequência, os demais algoritmos trabalham na criação de compassos com a série dodecafônica gerada, buscando dessa forma variações de combinações entre a duração das notas.

No decorrer deste trabalho a complexidade das restrições envolvidas é gradativamente aprimorada, obtendo assim resultados mais interessantes e mais complexos. Todos os algoritmos foram programados na linguagem *Fortran*, e os resultados obtidos foram transpostos para a simbologia musical pelo aplicativo *MuseScore*.

4.1 ALGORITMO 1

Objetivo: *Gerar uma série dodecafônica.*

Utilizando as notas da escala cromática deseja-se gerar uma sequência de maneira que duas notas consecutivas estejam no mínimo 1 e no máximo 3 tons de distância uma da outra. Dessa forma, é obtida uma série dodecafônica com características particulares quanto à distância entre duas notas consecutivas.

Aqui, há uma relação entre o problema exposto e o PVC visto anteriormente. Para fazer essa relação, cada nota musical é considerada uma cidade, e o objetivo é o de percorrer todas as 12 cidades, sem repetir nenhuma delas. A diferença é que aqui não há interesse em percorrer a menor distância possível, e sim a distância de 1 a 3 entre uma cidade e outra.

Estrutura: Primeiramente é solicitado ao usuário que forneça a solução inicial. Nesse momento, é considerado o vetor formado pelas 12 notas da sequência cromática partindo de Dó e subindo de meio em meio tom. Logo, pelo o foi exposto na Figura 16, os valores associados a solução inicial são 0,0, 0,5, 1,0 e assim por diante, até o 5,5. Dessa forma, fica completa a sequência que tem início em Dó e fim em Si.

A função objetivo é então representada pelas Equações 11 e 12.

$$F_{x_i} = \begin{cases} 1, se\ 1 \leq d_i \leq 3 \\ 0, c.c \end{cases}, \forall i = 1, \ldots, 11 \tag{11}$$

$$F_x = \sum F_{x_i} \tag{12}$$

Na equação 11, d_i representa a distância entre duas notas consecutivas. Dessa forma, d_1 equivale a distância entre a primeira e a segunda nota. Caso elas distem uma da outra um valor no intervalo [1,3], a função F_{x1} vale 1, caso contrário vale 0. A mesma verificação é feita para todas as trocas, já que i varia de 1 a 11, de modo que d_{11} é a distância entre a penúltima e a última nota. Logo, a partir da equação 11, a função F_{xi} analisa cada passagem entre as notas consecutivas do vetor solução, verificando se ela se enquadra ou não na condição estipulada. Se satisfizer a condição, é associado a ela o valor 1, caso contrário o valor 0. A equação 12 define a função fitness F_x, que é o somatório de todas as trocas da sequência do vetor x. É fácil perceber que o objetivo é maximizar tal valor, sendo encontrada a solução ótima quando ocorrer F_x=11, pois isso indicará que todas as trocas entre notas seguidas satisfazem a restrição estipulada inicialmente.

Com a solução inicial que foi dada, a função objetivo associada a ela possui valor nulo, já que todas as passagens entre notas consecutivas variam de 0,5 em 0,5, e tal valor não pertence ao intervalo [1,3]. Não atingindo o critério de parada (solução ótima), o algoritmo aplica o método *2-opt*, e a Figura 17 apresenta o passo a passo desse processo.

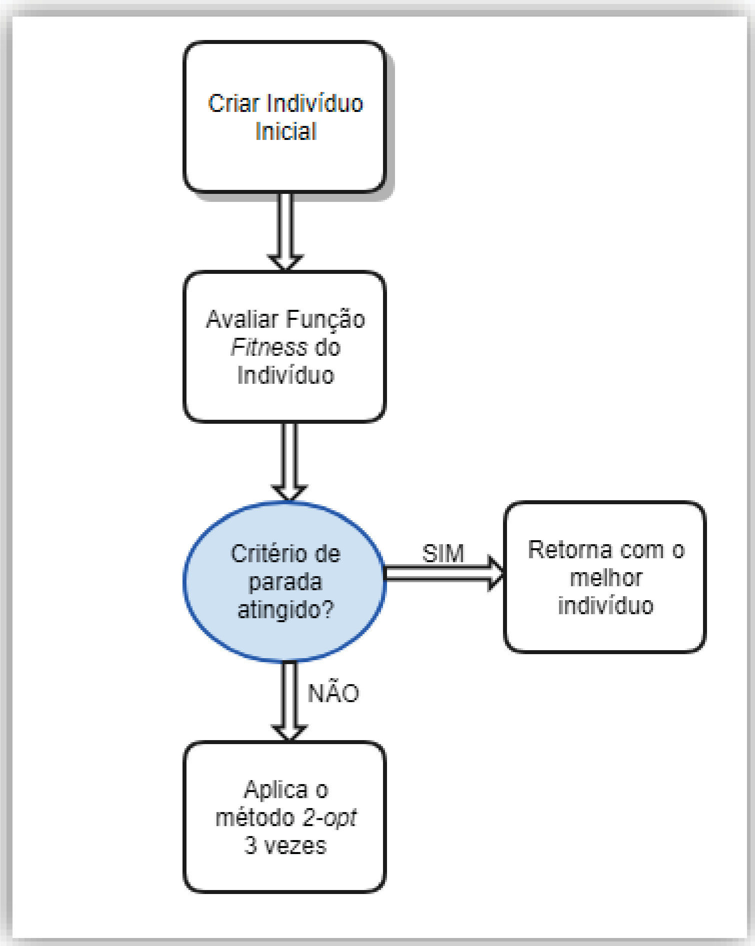

Figura 17: Passo 1 - Algoritmo 1

O método 2-opt é aplicado por três vezes, gerando três novas possíveis soluções. O método age, como foi visto, eliminando arcos da solução atual e religando-os, de modo a obter novas combinações. Por exemplo, se o vetor solução atual era X (representado na Equação 13) e sendo as posições de troca x_2 e x_7 definidas aleatoriamente, ficaríamos com o vetor X*, representado na Equação 14.

$$X = (C; \boldsymbol{C}\#; D; D\#; E; F; \boldsymbol{F}\#; G; G\#; A; A\#; B) \qquad (13)$$

$$X = (C; \boldsymbol{F}\#; D; D\#; E; F; \boldsymbol{C}\#; G; G\#; A; A\#; B) \qquad (14)$$

Uma vez que as 3 combinações foram geradas e que suas funções fitness foram calculadas, basta compará-las e decidir se o vetor inicial é mantido ou substituído por alguma dessas trocas. Caso nenhuma delas tenha apresentado melhora, descartam-se os novos indivíduos e a solução inicial é mantida. Caso contrário, a solução que apresentou melhor valor da função fitness assume o lugar da solução inicial, como mostra a Figura 18.

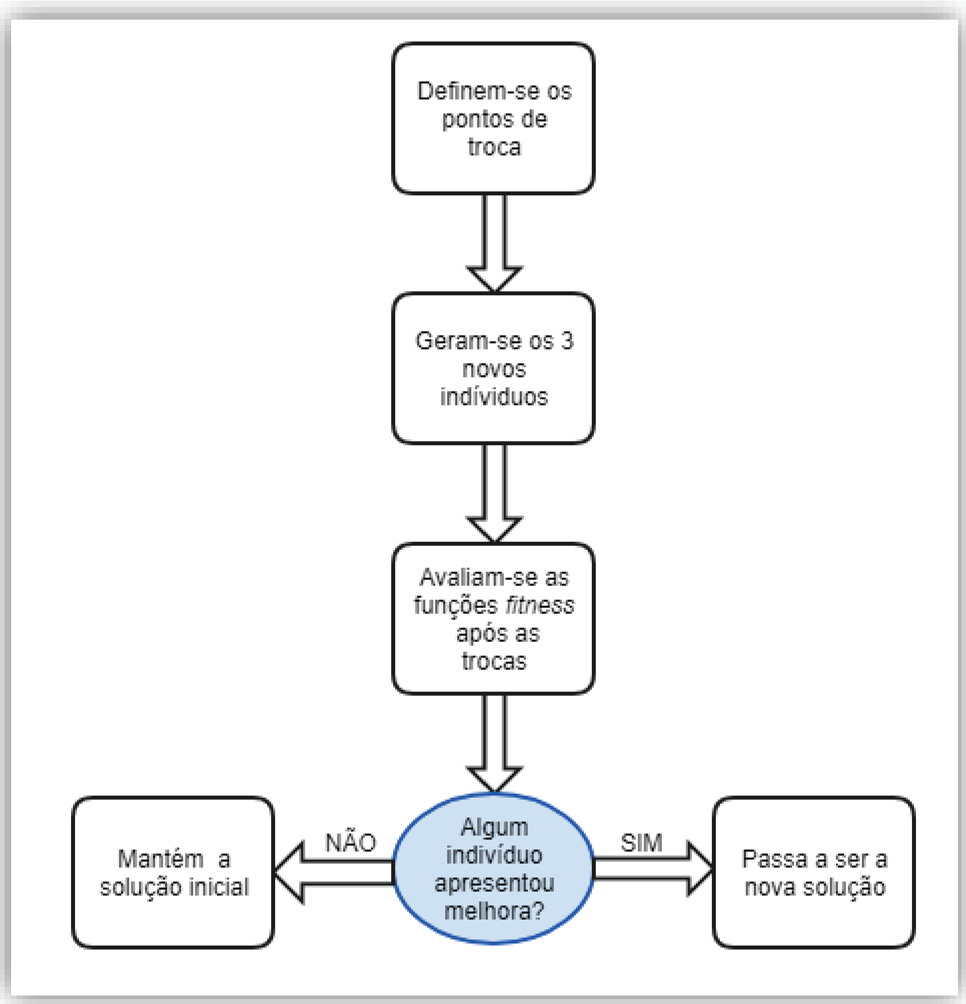

Figura 18: Aplicação do método *2-OPT*

.

Sendo definida uma nova solução, o ciclo volta ao Passo 1 apresentado na Figura 17e assim segue, até que atinja o critério de parada e retorne a solução ótima.

Diversas são as formas de definir o critério de parada em uma programação, e aqui foram implementados dois deles. O primeiro diz respeito ao ciclo inicial, que foi criado para efetuar 50 iterações, de modo que ao final delas o algoritmo apresenta a solução obtida, podendo ser uma solução ótima ou não. No entanto, outro critério foi implementado no meio do processo, e ele diz respeito a estagnação da solução por iterações consecutivas. Assim sendo, caso não haja melhora da função fitness por 5 iterações consecutivas, o algoritmo irá finalizar e retornar a solução em questão.

Para realizar esse controle é inserida no algoritmo a variável W, que inicia valendo 0 e é atualizada a cada vez que a solução não obtém melhora. Cada vez que isso ocorrer, W passa a valer uma unidade a mais do que valia, e sempre que uma nova solução é escolhida W volta a valer 0. Dessa forma, quando W for igual a 5 o programa encerra, retornando a solução obtida. A Figura 19 ilustra esse critério de parada.

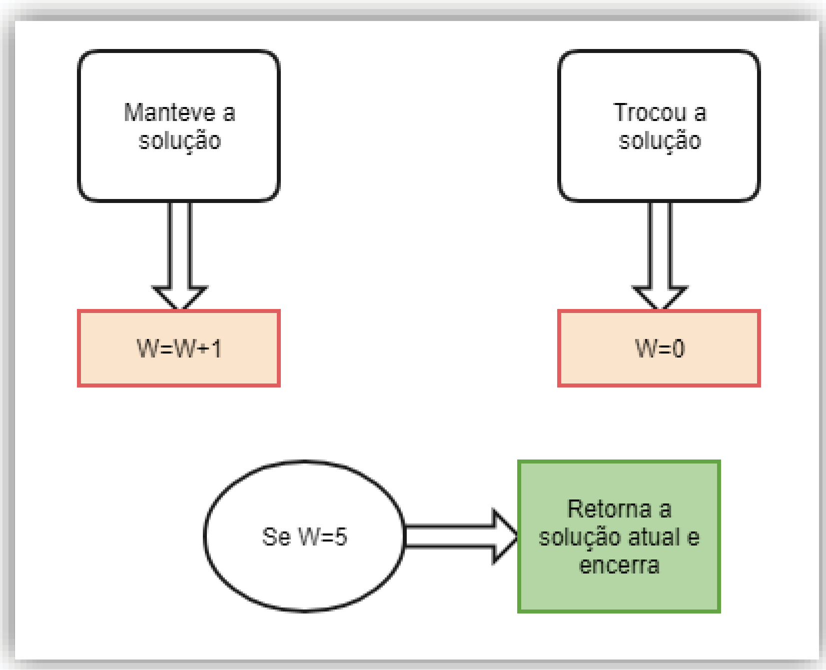

Figura 19: Critério de parada - Algoritmo 1

Repetindo o ciclo até que algum dos critérios de parada seja satisfeito, a solução obtida é apresentada, e os resultados são apresentados a seguir.

Resultados: Para a solução inicial fornecida, o resultado obtido é apresentado na Figura 20, que relaciona os valores às notas musicais associadas, gerando a sequência dodecafônica correspondente.

$X_1 = 0.0$	$x_2 = 3.0$	$x_3 = 4.5$	$x_4 = 2.0$
$X_5 = 0.5$	$x_6 = 1.5$	$x_7 = 3.5$	$x_8 = 5.0$
$X_9 = 4.0$	$x_{10} = 1.0$	$x_{11} = 2.5$	$x_{12} = 5.5$

C F# A E C# D# G A# G# D F B

Figura 20: Solução do algoritmo 1

Observações: Embora tenha-se chegado a uma solução que satisfaz plenamente a condição imposta, há uma limitação que deve ser citada, e ela se refere à função que gera valores aleatório no software Fortran, que por serem pseudoaleatórios acabam fornecendo os mesmos números, retornando assim a mesma sequência de trocas, independente de quantas vezes o programa seja rodado. Isso limita o trabalho, pois partindo da mesma solução inicial, chega-se sempre a mesma solução ótima. Assim sendo, um dos maiores desafios nos algoritmos desenvolvidos a seguir é o de contornar esse problema, aumentando a diversidade das soluções obtidas.

4.2 ALGORITMO 2

Objetivo: *Agrupar a sequência obtida no algoritmo 1 em 4 compassos 2 por 4.*

Para isso, são utilizadas inicialmente duas figuras rítmicas, a semínima e a colcheia, de modo que o valor 1 está associado à primeira e o valor 0,5 à segunda. Isto porque, como visto anteriormente, em um compasso com denominador 4 a semínima equivale a uma unidade de tempo, enquanto a colcheia equivale a meia unidade.

Note que nesse caso é possível uma associação do problema exposto com o Problema da Mochila Compatimentada. Para tanto, basta relacionar cada nota a um objeto e cada figura rítmica ao peso desse objeto. Cada compasso é um bolso da mochila, e a fórmula de compasso define a capacidade de cada bolso. O objetivo é então o de preencher a mochila com os objetos disponíveis, respeitando as capacidades envolvidas.

Diferentemente do PMC tradicional, o peso aqui não é previamente conhecido. Tal valor é variável, e o objetivo é o de buscar a melhor combinação entre os pesos, de modo a poder carregar todos os objetos. Dessa forma, utilizam-se todas as notas musicais da sequência e o compasso fica plenamente preenchido.

A metaheurística utilizada na busca do valor ótimo é Algoritmos Genéticos e utiliza dois operadores, primeiramente o cruzamento e depois a mutação.

Estrutura: Diferentemente do algoritmo anterior, no qual criou-se manualmente a solução inicial, aqui tal solução é gerada de maneira aleatória. Para isso, utiliza-se a função do próprio software *Fortran* para criar os dois indivíduos iniciais X e Y. Foi comentado anteriormente que tal função é limitada por gerar pseudoaleatórios, e por essa razão o algoritmo 3 usa a estrutura deste, mas gera os vetores utilizando outra ferramenta, o que permite a comparação dos resultados.

Retornando para o problema atual, como o comando retorna valores aleatórios entre 0 e 1 é necessário que se verifique se o número gerado representa uma semínima ou uma colcheia. Sejam ent A e B os vetores aleatórios gerados, de modo que A e B possuem 12 elementos cada um. O vetor A é responsável por gerar o indivíduo inicial X, e sendo A= {a_1, a_2, ..., a_n} e X= {x_1, x_2, ..., x_n}, a Figura 21 ilustra como a população inicial é gerada.

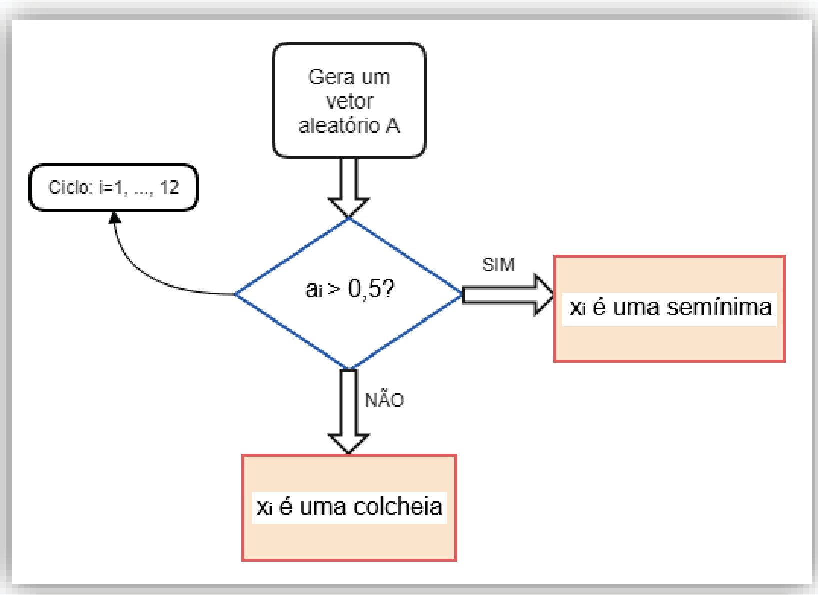

Figura 21: Indivíduo inicial X

Da mesma forma, sendo B= $\{b_1, b_2, ..., b_n\}$ e Y= $\{y_1, y_2, ..., y_n\}$ e seguindo o passo a passo ilustrado na Figura 21, obtém-se o vetor Y.

Como deseja-se distribuir para cada objeto um peso, de modo que a capacidade de cada bolso seja completamente utilizada, nada mais natural e lógico que se analise o preenchimento de cada bolso, verificando se ele está ou não completo. A Figura 22 ilustra o preenchimento do bolso (compasso) 1, onde k_i, com i=1, ..., 12, representa os elementos do vetor k e $t_{(1, k)}$ agrupa os elementos k_i enquanto o valor é inferior a 2, que é a capacidade do bolso. Uma vez que esse valor excede a capacidade, a variável $u_{(1, k)}$ é definida, e representa o último objeto pertencente ao primeiro bolso. Não excedendo o valor 2, o algoritmo segue agrupando novos objetos, até que o critério seja satisfeito, e tal processo é ilustrado pela Figura 22.

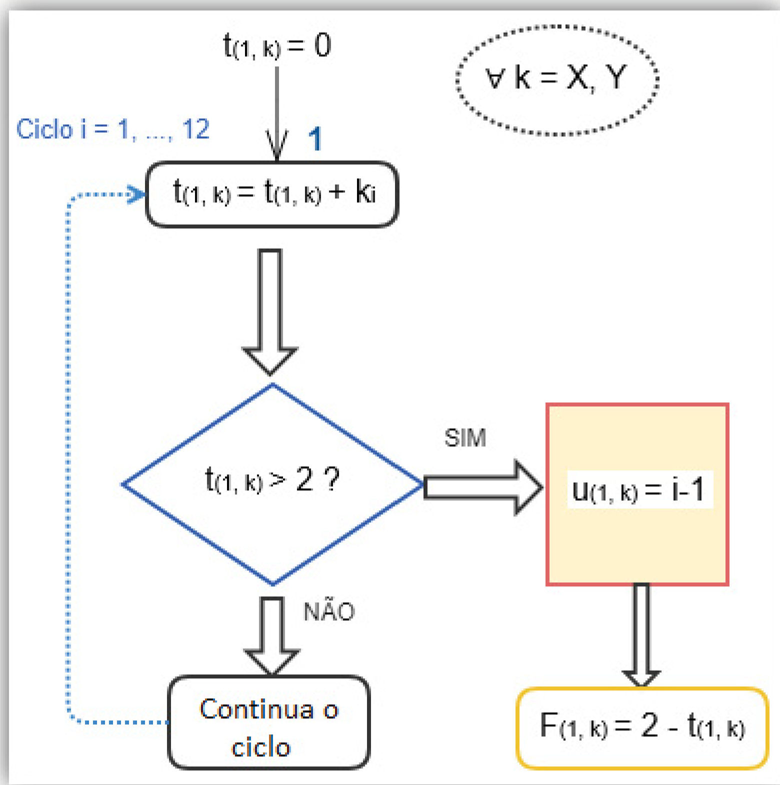

Figura 22: Formando o primeiro compasso

Uma vez fechado o compasso é preciso verificar se as figuras rítmicas que o integram o completam ou não, visto que se faz necessário que elas somem 2. A função fitness $F_{(1, k)}$ associada ao primeiro compasso faz essa análise, calculando a falta associada ao primeiro compasso dos vetores X e Y. Além disso, uma vez executado o passo descrito na Figura 22 para o primeiro compasso dos vetores da população inicial é necessário aplicá-lo aos demais compassos, formando o segundo, o terceiro e o quarto. Sendo $u_{(j,\ k)}$ o último termo do j-ésimo compasso do vetor k, o compasso seguinte inicia da posição $u_{(j,\ k)}$ +1, sendo necessário que se verifique se ela ainda pertence ao intervalo [1,12], caso contrário todas as notas já foram distribuídas pelos compassos anteriores e função fitness $F_{(j,k)}$ associada a esse compasso será igual a 2, já que o compasso encontra-se vazio.

A Figura 23 apresenta esse processo de verificação, e caso restem notas a serem distribuídas, o algoritmo retoma ao passo 1 indicado na Figura 22.

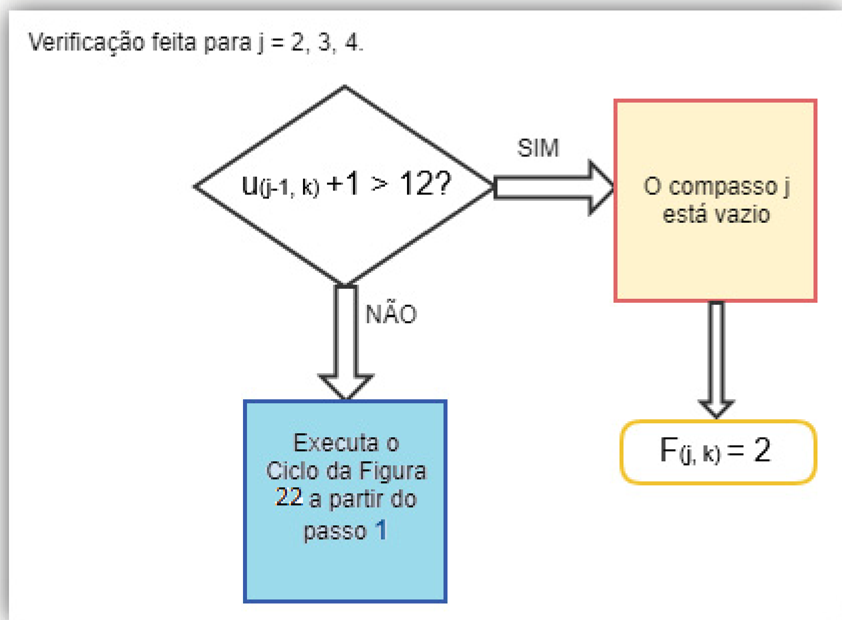

Figura 23: Completando os demais compassos

Se por um lado pode ocorrer de algum compasso ficar vazio, por outro há a possibilidade de que ao final dos 4 compassos nem todas as notas tenham sido utilizadas, havendo assim uma sobra S, cujo valor deve ser incorporado as funções fitness dos compassos, já que representa também uma inexatidão. A equação 15 analisa a função objetivo da população inicial, sendo F_x a função fitness do vetor X, e F_y a função fitness do vetor Y.

$$F_k = S_k + \sum_{j=1}^{4} F_{(j,k)}\,, \forall\ k = X, Y \tag{15}$$

O objetivo é o de minimizar a função F_k, garantindo assim a busca pela distribuição exata das 12 notas nos 4 compassos.

Com a população inicial definida e estando as funções fitness calculadas, parte-se para a aplicação da metaheurística, cujo objetivo é gerar cruzamentos dos indivíduos iniciais, analisar esses cruzamentos e selecionar os que apresentarem melhores resultados. Para isso, aplica-se o método do cruzamento entre dois pontos, que são definidos aleatoriamente, gerando dois novos indivíduos Z1 e Z2, cujas funções objetivos também devem ser calculadas, como mostra a Figura 24.

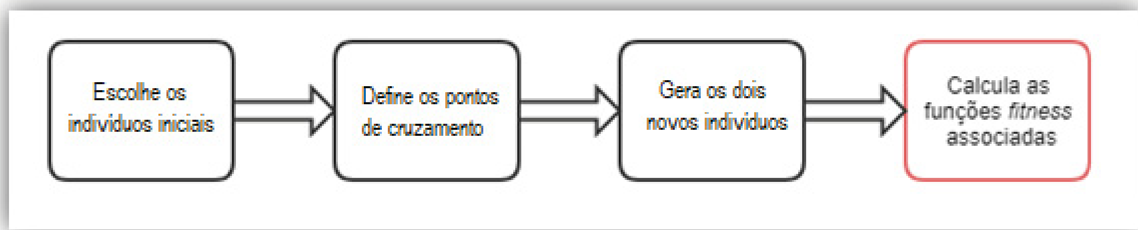

Figura 24: Cruzamento (Crossover)

Tendo agora a população composta pelos 4 indivíduos e as funções fitness F_x, F_y, F_{z1} e F_{z2} associadas, deve-se selecionar os dois melhores adaptados que irão compor a população inicial, e a solução ótima é obtida quando F_k for igual a zero, para algum $k \in$ população. Caso a condição seja satisfeita, o algoritmo encerra e retorna o vetor solução, caso contrário ele segue com os cruzamentos, como mostra a Figura 25.

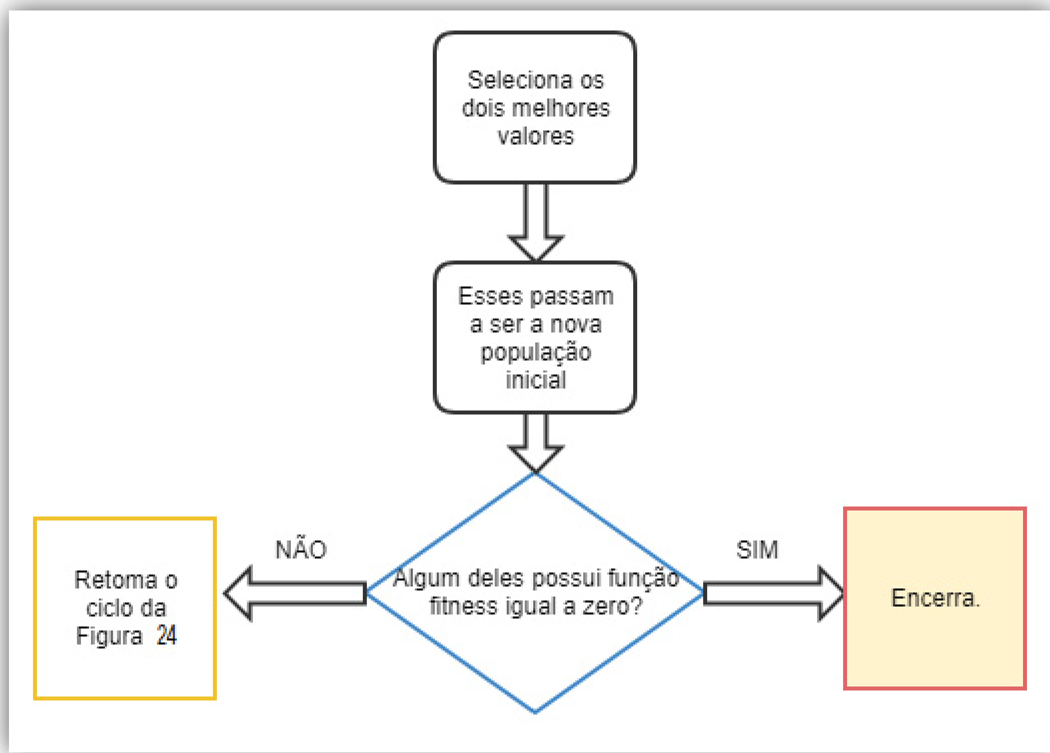

Figura 25: Seleção dos melhores adaptados

O quadro amarelo da Figura 25 indica que enquanto o ótimo não for encontrado o ciclo continua cruzando os indivíduos e analisando as populações, baseando-se na seleção natural, que é a característica principal do método.

No entanto, pode ocorrer em algum momento a perda da diversidade, uma vez que a população possui um número pequeno de indivíduos. A consequência disso é que os pais serão muito semelhantes entre si, e os cruzamentos também o serão, e assim sendo, não ocorrerá mais melhora na função objetivo. A Figura 26 apresenta um comando que verifica se os valores das funções fitness dos indivíduos da população são iguais, identificando assim a perda de diversidade, passo que interromperá o ciclo de cruzamentos e passará para o ciclo de mutações.

.

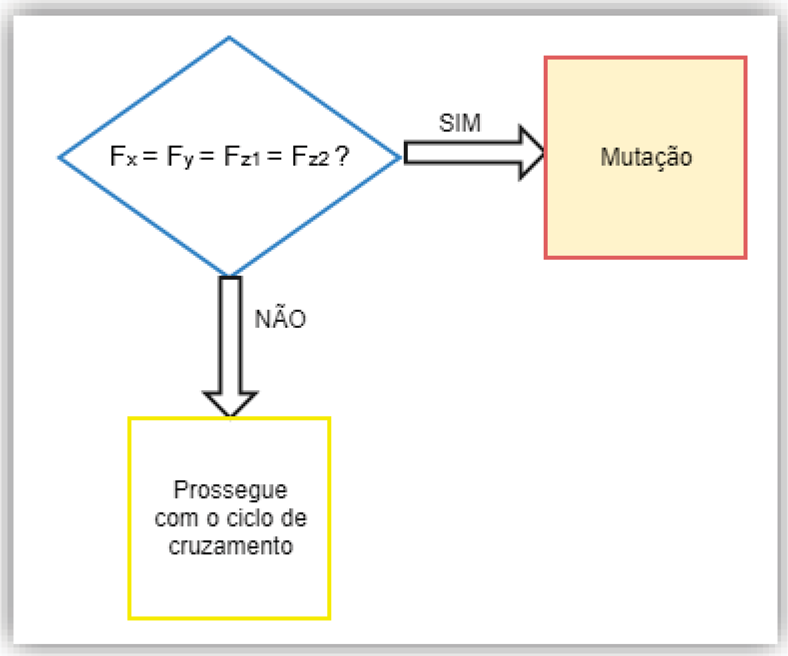

Figura 26: Perda de diversidade

Para a implementação da mutação, que será aplicada em dois genes do indivíduo, calcula-se de forma aleatória as posições *i* e *j* que sofrerão alteração, e a mutação age no gene invertendo seu valor. Se a posição x_k (para k=i, j) valia 0,5 passa a valer 1,0, e se valia 1,0 passa a valer 0,5, e o passo a passo dessa verificação pode ser observado na Figura 27.

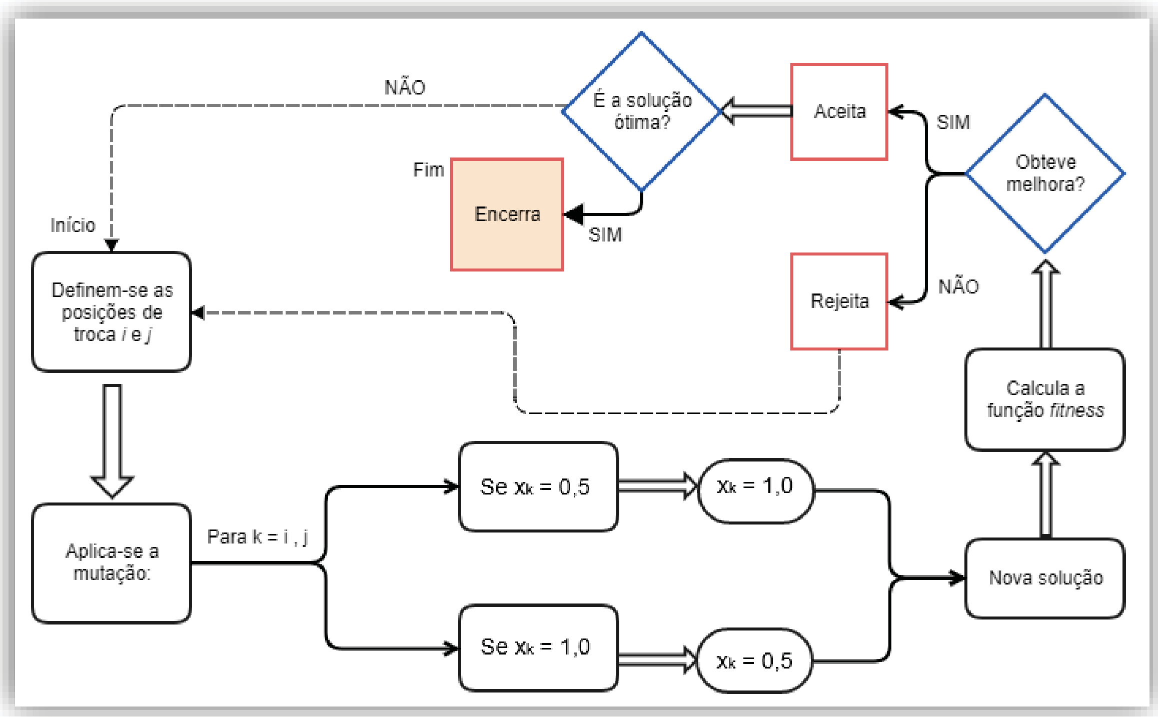

Figura 27: Mutação - Algoritmo 2

Aplicada a mutação nos dois pontos definidos, o algoritmo compara a função objetivo da solução anterior com a nova, e a que tiver menor valor será escolhida como nova solução. Em seguida verifica-se novamente se o valor ótimo foi obtido, e em caso afirmativo o programa encerra e a solução é apresentada. Caso contrário, é dado sequência ao ciclo de mutações, seguindo os passos apresentados pela Figura 27.

Resultados: Após aplicar o algoritmo 2 chega-se a solução apresentada na Figura 28.

x_1=0.5	x_2=0.5	x_3=1.0	x_4=0.5
x_5=1.0	x_6=0.5	x_7=1.0	x_8=0.5
x_9=0.5	x_{10}=0.5	x_{11}=0.5	x_{12}=1.0

Figura 28: Solução do algoritmo 2

Note que os valores obtidos representam figuras rítmicas, de modo que cada uma está associada a uma nota da sequência gerada no algoritmo 1, e a Figura 29 mostra como fica a disposição nos 4 compassos desejados.

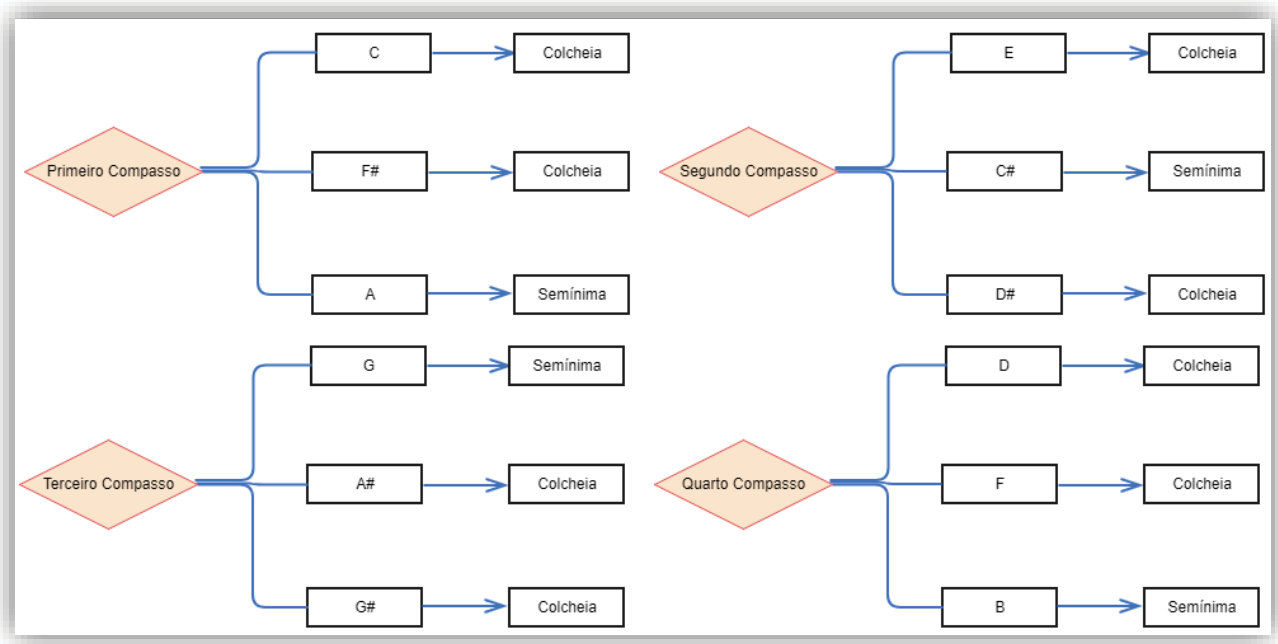

Figura 29: Compassos formados pelo algortimo 2

E a Figura 30 mostra a solução obtida em forma de partitura

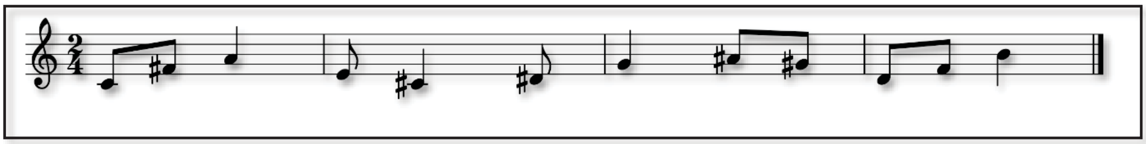

Figura 30: Solução obtida pelo algoritmo 2 na partitura

4.3 ALGORITMO 3

Objetivo: *Alterar o método de geração de aleatórios utilizada no algoritmo 2.*

A diferença principal entre o algoritmo 2 e este é que agora a população inicial não será mais gerada através do comando do software *Fortran*, e sim pelo software *Excel*, de modo que o programa lê os dados de um arquivo externo. Além disso, os aleatórios utilizados nos operadores genéticos são gerados a partir de dados fornecidos pelo próprio usuário ao programa. Desse modo, evita-se a repetição trazida pelos pseudoaleatórios e cria-se maior variedade na busca por soluções ótimas.

Estrutura: A estrutura do algoritmo 2 para o 3 difere em pontos específicos, pois a maioria dos comandos entre eles são semelhantes. A primeira diferença surge logo no início, quando é solicitado ao usuário que insira os indivíduos iniciais da população. No algoritmo 2, esse passo era feito de maneira automática pela função do próprio programa *Fortran* que gera números aleatórios. Já neste programa tais valores são gerados utilizando a ferramenta *Excel*.

A estrutura segue de maneira semelhante a anterior, onde são analisadas as funções fitness da população inicial, e sofre uma alteração na sequência, no momento em que são definidos os pontos de cruzamento entre os pais. Essa mudança ocorre porque, novamente, não se faz uso do gerador de números aleatórios. Para gerar os aleatórios é solicitado ao usuário que forneça os minutos e segundos atuais, e a partir desses valores as posições de cruzamento são calculadas. Tal procedimento garante melhor desempenho dos números gerados, já que o comando anterior limitava a busca pelas soluções, e esse a cada instante gera novos valores. Fornecidos pelo usuário os dados, o cálculo dos aleatórios a_1 e a_2 ocorre como mostram as Equações 16 e 17, obtendo dessa forma as posições de cruzamento.

$$a_1 = \left(\frac{min+50}{seg+50}\right) \cdot 12 \tag{16}$$

$$a_2 = \left(\frac{min+10}{seg+10}\right) \cdot 12 \tag{17}$$

. Nos demais passos os algoritmos coincidem, e pode-se analisar os resultados obtidos a seguir.

Resultados: Ao fornecer os vetores iniciais gerados aleatoriamente por um meio externo há ganho de diversidade na população inicial. Logo, isso permitiu maior variedade nas soluções obtidas, e as Figuras 31, 32 e 33 apresentam os resultados, tanto em formato vetorial como representados na partitura.

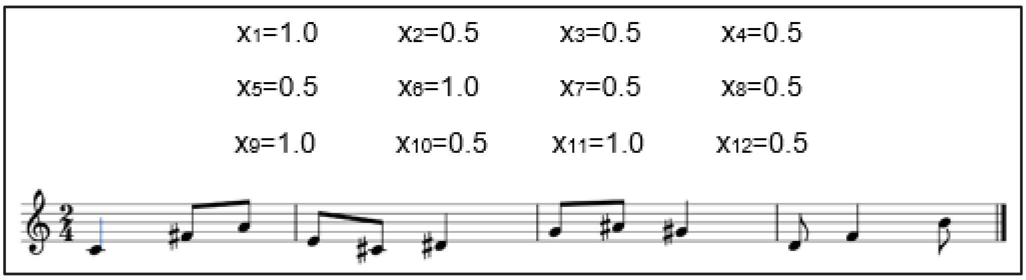

Figura 31: Primeira solução do algoritmo 3

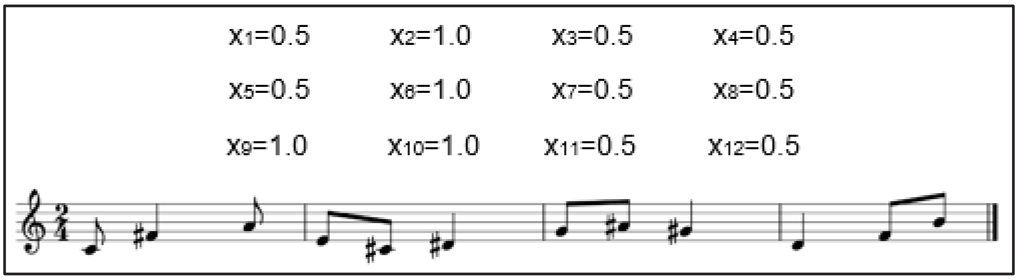

Figura 32: Segunda solução do algoritmo 3

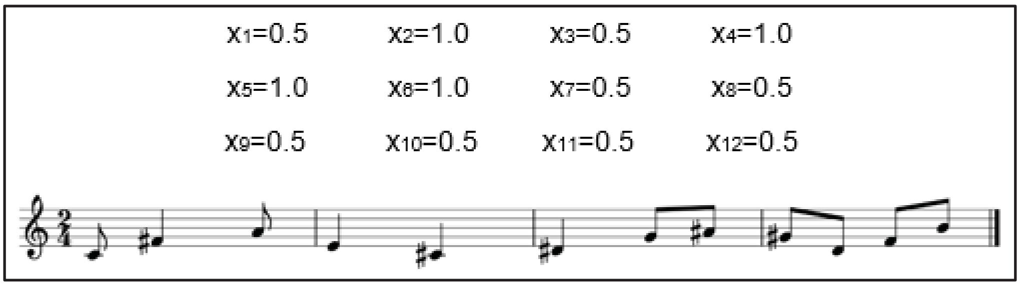

Figura 33: Terceira solução do algoritmo 3

4.4 ALGORITMO 4

Objetivo: *Agrupar a sequência obtida no algoritmo 1 em 2 compassos 3 por 4 e 2 compassos 5 por 4.*

O objetivo é dar mais diversidade aos compassos criados, alternando entre formatos e mostrando, dessa forma, que várias combinações são possíveis. Além disso, conta-se também com maior variedade de figuras rítmicas, fazendo uso agora da semicolcheia, colcheia, semínima e mínima. Como antes, cada figura rítmica tem um valor numérico associado, e os valores escolhidos se justificam pela fórmula dos compassos, que possuem denominador igual a 4. A Figura 34 apresenta a relação entre as figuras e seus respectivos valores.

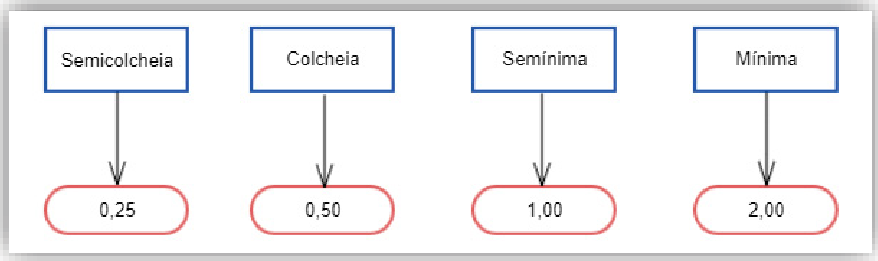

Figura 34: Valores das figuras rítmicas no algoritmo 4

O problema em questão é semelhante ao abordado no algoritmo 2, com a diferença que agora a combinação de fórmulas de compasso é distinta. Da mesma forma que antes, o objetivo é o de preencher cada bolso (compasso) com os objetos (notas) disponíveis, atribuindo pesos aos objetos de modo que seja possível preencher os bolsos de acordo com sua capacidade ao mesmo tempo que se usam todos os objetos disponíveis.

Estrutura: Pode-se basear na mesma estrutura aplicada aos algoritmos anteriormente desenvolvidos, já que possuem a mesma estrutura lógica. No entanto, o aumento do conjunto das figuras rítmicas utilizadas traz uma diferença quanto a forma de gerar a população inicial, que ocorre como mostra a Figura 35.

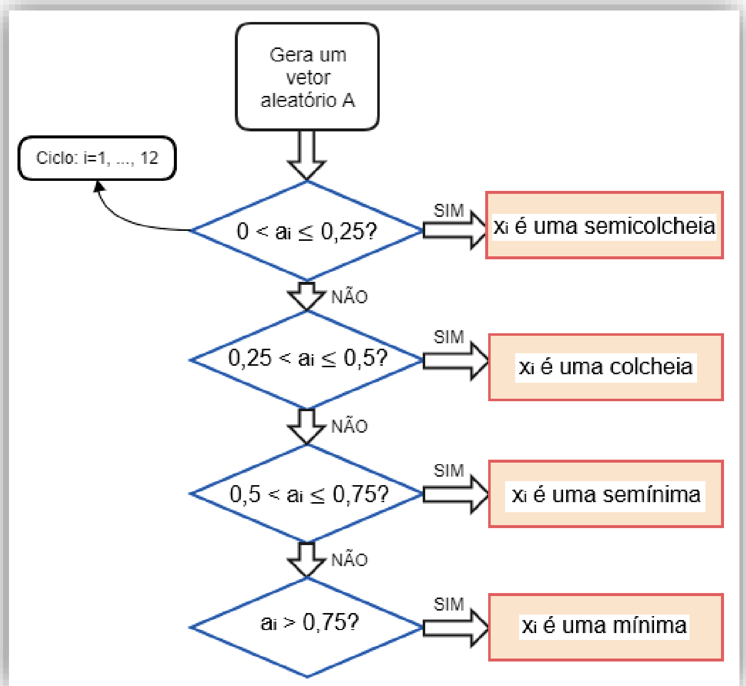

Figura 35: População Inicial – Algoritmo 4

Na Figura 35, A é o vetor aleatório que gera o indivíduo X. Para gerar Y basta repetir o processo para o vetor aleatório B.

Sendo a população inicial conhecida, verifica-se o valor da função fitness de cada um de maneira semelhante ao que foi feito no algoritmo 2. Os passos seguintes de aplicação da metaheurística também se repetem, pois são mantidos os dois pontos de cruzamento e a geração da população composta por tais cruzamentos. No entanto, embora os passos ocorram de maneira análoga a anterior, um passo adicional é inserido ao final desse processo, destinado a identificar a perda de diversidade nas gerações, como mostra a Figura 36.

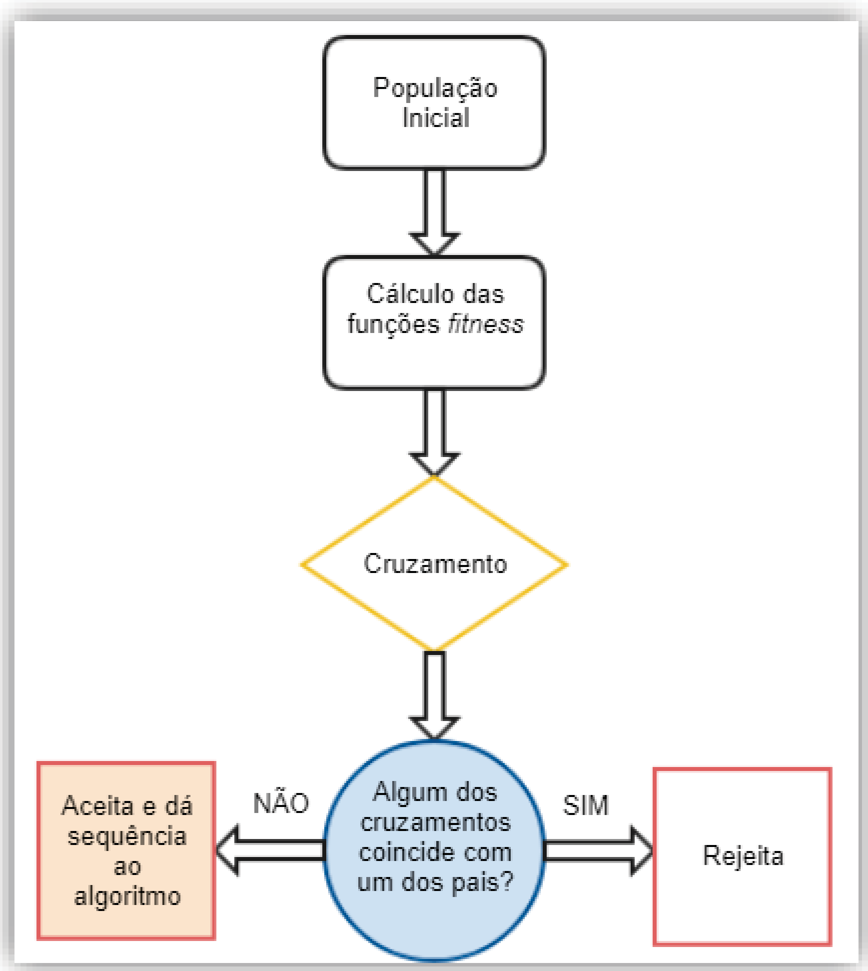

Figura 36: Controle da perda de diversidade

A sequência de passos descrita na Figura 36 faz com que se rejeitem os cruzamentos que são semelhantes aos pais, pois sendo aceitos não trariam diversidade à população. Caso sejam distintos, aceitam-se os cruzamentos e analisam-se suas funções objetivos, seguindo a mesma estrutura de antes. Tendo calculado os valores das funções, comparam-se os resultados e escolhem-se os dois melhores adaptados. Após essa seleção, caso a solução ótima seja obtida, o programa encerra. Caso contrário, seguem-se os cruzamentos, até que por 5 iterações seguidas não haja melhora (Figura 19). Se isso ocorrer antes da solução ótima ter sido encontrada, a mutação é implementada e segue a estrutura apresentada no algoritmo 2, exceto na definição dos valores de troca, que aqui ocorre como mostra a Figura 37.

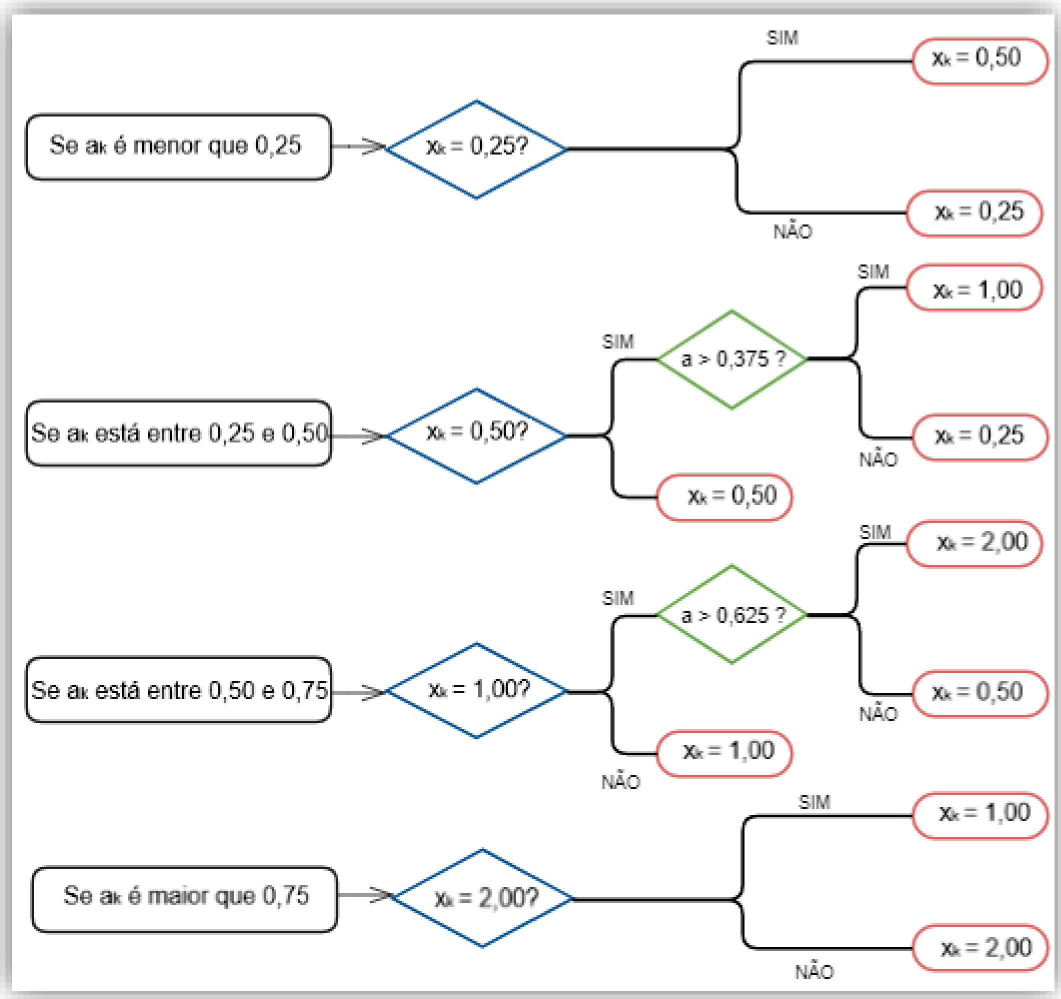

Figura 37: Mutação - Algoritmo 4

Na Figura 37, a_k é o número aleatório gerado, e x_k é a posição na qual a mutação é aplicada, com k= i,j. O que essa sequência de passos e verificações faz é definir qual a mutação feita no ponto definido, já que há 4 possíveis valores, e não 2 como antes. Assim sendo, dividindo o intervalo [0,1] em quatro partes, a cada parte é associada uma figura rítmica. Por exemplo, se o aleatório gerado é menor que 0,25, é associado a ele a semicolcheia. No entanto, pode ser que essa posição já seja originalmente ocupada pela figura rítmica destinada, e sendo esse o caso, a figura será trocada por uma colcheia. Da mesma forma ocorre para os demais valores possíveis de a_k, sempre analisando qual é a figura atual, para garantir que a mutação não provocará efeito nulo, que ocorreria caso uma figura rítmica fosse trocada por outra de mesmo valor.

Definidos os pontos e aplicada a troca das coordenadas analisa-se a nova solução e baseando-se no valor de sua função fitness opta-se por continuar com a solução anterior ou substituí-la pela nova. E assim a busca segue, até que retorne o vetor que possui o valor objetivo desejado ou então até que se esgote o número máximo de iterações.

Resultados: A população inicial composta pelos vetores X e Y é representada pelas Equações 18 e 19.

$$X = (2.00; 1.00; 0.25; 1.00; 0.50; 1.00; 0.50; 1.00; 1.00; 1.00; 0.50; 0.25) \qquad (18)$$

$$Y = (0.50; 2.00; 1.00; 0.50; 1.00; 1.00; 2.00; 0.25; 0.50; 2.00; 2.00; 2.00) \qquad (19)$$

E, após a aplicação da metaheurística retornou o vetor solução X* representado pela Equação 20.

$$X^* = (2.00; 0.50; 0.50; 2.00; 1.00; 1.00; 2.00; 1.00; 0.50; 1.00; 2.00; 2.00) \qquad (20)$$

A função objetivo do vetor indicado na Equação 20 equivale a 0,50, pois o terceiro compasso não fecha exatamente em 5, e sim em 4,5. Logo, o procedimento não chegou à solução ótima desejada.

4.5 ALGORITMO 5

Objetivo: *Aperfeiçoar o algoritmo 4 para que a solução ótima possa ser obtida.*

Como mostrou a Equação 20, o algoritmo 4 retornou uma solução que não corresponde a todas as restrições do problema. O objetivo agora é o de contornar esse problema, buscando novas implementações e formas de resolvê-lo.

Para isso, o problema é novamente modelado como um PMC, mas agora ao aplicar a mutação analisa-se a função fitness não mais de maneira geral (a partir de F_x) e sim a partir dos compassos ($F_{(1,\ x)}$, $F_{(2,x)}$, ...), buscando assim a solução que melhor se adapta a cada um deles, partindo do primeiro e indo até o quarto.

Como visto anteriormente, no problema da mochila compartimentada cada bolso tem uma capacidade diferente, e é exatamente o que ocorre aqui ao trabalharmos com fórmulas de compasso distintas. O algoritmo 5 trabalha na busca de uma solução ótima selecionando os indíviduos mais bem adaptados em um determinado compasso, não mais comparando as funções fitness totais.

Estrutura: Uma vez que o objetivo é buscar a melhora da função objetivo não mais de maneira geral, mas por etapas, a escolha do melhor indivíduo não toma como base a sobra total deixada, mas sim a sobra em cada compasso, de modo que o algoritmo inicia buscando a solução que apresenta o melhor desempenho no primeiro compasso. A Figura 38 ilustra como ocorre a seleção.

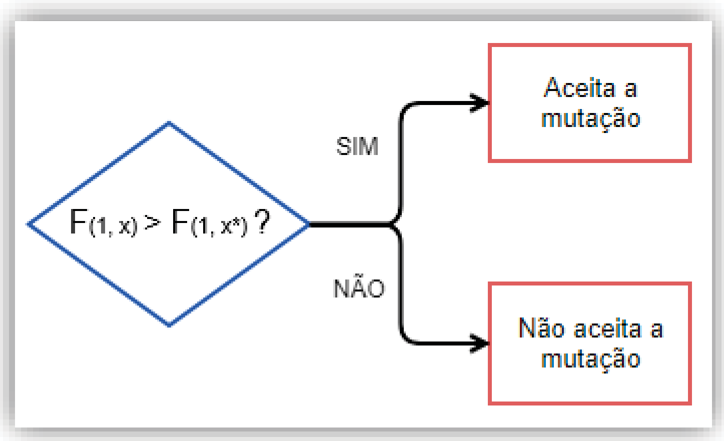

Figura 38: Mutação - Verificação do primeiro compasso

Ao aplicar a mutação, considera-se X como a solução anterior a mutação e $F_{(1,\ x)}$ a função fitness dessa solução no primeiro compasso. X* é o vetor posterior a mutação, sendo $F_{(1,\ x^*)}$ a função fitness associada a ele no primeiro compasso. O que os passos descritos na Figura 38 fazem é analisar se a mutação implicou ou não na melhora do primeiro compasso, e em caso afirmativo o vetor X* passa a ser a nova solução X.

Estando completo o primeiro compasso o mesmo ocorre no segundo e assim por diante, até que os quatro compassos sejam preenchidos, sempre seguindo a estrutura apresentada na Figura 38.

Dessa forma parte do problema é resolvido, já que com os compassos sendo completados individualmente e em crescente obtém-se uma melhor eficácia. Por outro lado, ao executar o programa, um problema de infactibilidade surge, pois no terceiro compasso todas as notas acabam sendo utilizadas, não restando nenhum objeto para o último bolso, que por sua vez fica vazio. Para contornar esse problema, os intervalos do ciclo responsável pela formação dos compassos devem ser alterados, garantido que haja sempre o valor mínimo de notas necessárias em cada um deles. Partindo da ideia proposta pela Figura 22, que define o último termo $u_{(1,\ k)}$ do primeiro compasso e permite a generalização de $u_{(j,k)}$ para o último termo do compasso j no vetor k, a Figura 39 implementa uma restrição com relação a esse termo.

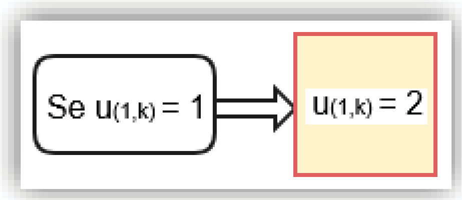

Figura 39: Restrição sobre o último termo do primeiro compasso

Esse simples passo garante que o último termo do primeiro compasso seja no mínimo igual a 2. Isso porque em um compasso 3 por 4, onde a figura rítmica de maior valor é a mínima – que equivale a duas unidades de tempo – são necessárias, no mínimo, duas figuras para fechar o valor 3. De maneira análoga ocorre no segundo compasso, cuja fórmula também é 3 por 4. Já nos compassos 3 e 4 necessitam-se de no mínimo 3 notas em cada um para garantir que será possível somar o valor 5, e o ciclo fica definido como mostra a Figura 40.

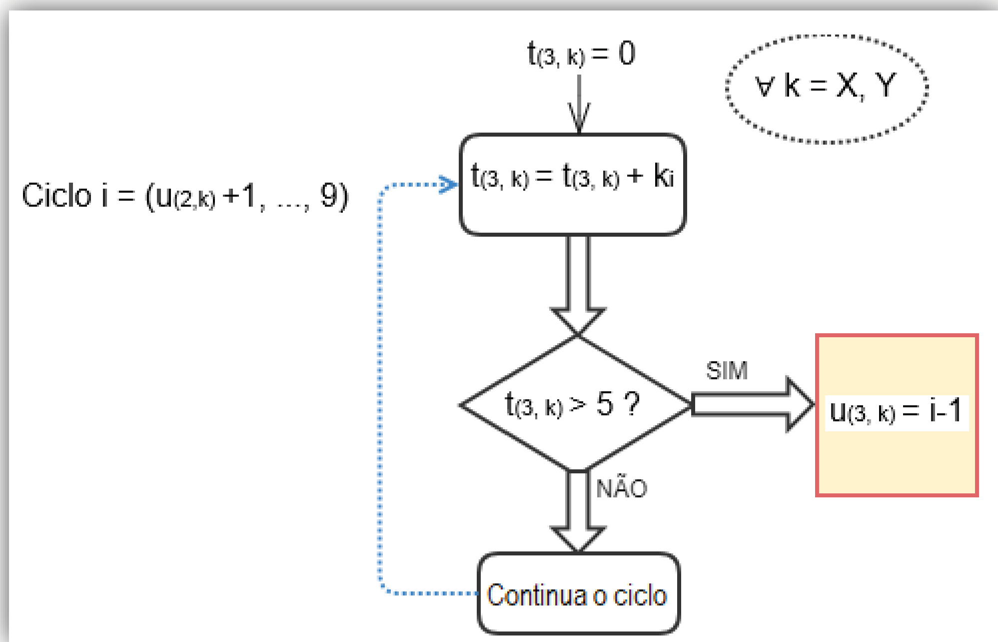

Figura 40: Restrição sobre o último termo do terceiro compasso

Os passos descritos na Figura 40 garantem que restem no mínimo 3 notas para o quarto compasso, já que i vai no máximo até 9, dissolvendo assim o problema de infactibilidade.

Antes de encerrar as comparações entre o algoritmo anterior e esse, resta dizer que um passo foi inserido no cálculo dos números aleatórios. Isso ocorreu porque estavam sendo gerados com uma frequência muito grande números racionais menores que 0,1, o que implicava na escolha da mesma figura rítmica sempre. Para contornar esse problema é que surge a restrição imposta pela Equação 21, garantindo que se o aleatório a_k gerado for menor que 0,1, ele será multiplicado por 10.

$$Se\ a_k < 0{,}1 \ \Rightarrow a_k = a_k \times 10 \qquad (21)$$

Nos demais comandos as estruturas coincidem e pode-se observar os resultados obtidos a seguir.

Resultados: Partindo da mesma solução inicial imposta ao algoritmo 4, chega-se a solução apresentada na Figura 41.

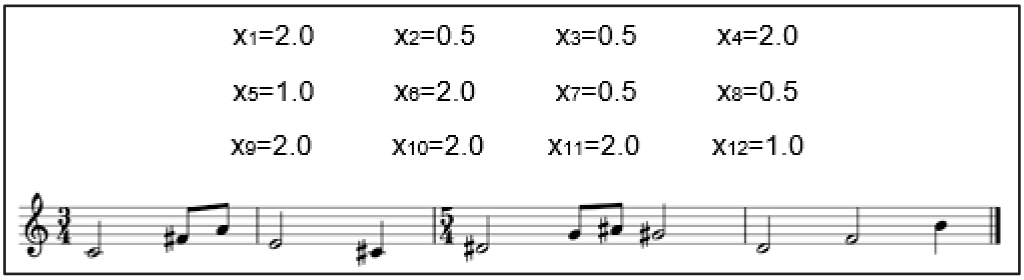

Figura 41: Solução obtida através do algoritmo 5

4.6 ALGORITMO 6

Objetivo: *Agrupar a sequência de notas em dois compassos 2 por 4 e dois compassos 2 por 2.*

Nesta última programação desenvolve-se algo mais complexo, já que os denominadores das fórmulas de compasso são distintos, fato que altera toda a relação de “tamanho” entre as figuras rítmicas. No compasso com denominador 4 – o qual foi explorado até agora – a semínima é a figura que representa a unidade. Já no compasso com denominador 2 a figura que tem essa característica é a mínima.

Para facilitar a compreensão, tais valores podem ser observados na Figura 42.

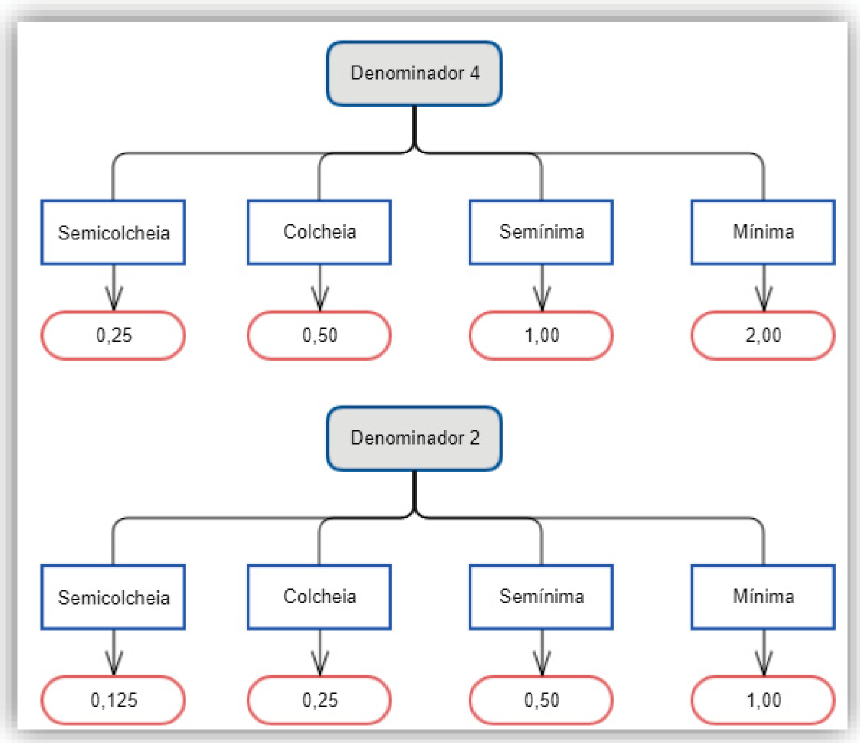

Figura 42: Valores das figuras rítmicas - Algoritmo 6

Com relação à estrutura, essa se mantém semelhante às anteriores, exceto na atribuição dos valores dados a cada figura, que como mostra a Figura 42 será diferente entre um compasso e outro.

Estrutura: Da mesma forma que antes, os vetores iniciais são gerados externamente, mas aqui possuem aqui uma diferença. Como a mínima possui um valor de duração alto se comparada as demais figuras, e os numeradores dos compassos são números baixos, é necessário atribuir um peso menor à ela, para que a probabilidade de ocorrer uma mínima seja menor. Já a semicolcheia possui um valor baixo, além de não ter integrado a solução ótima encontrada pelo algoritmo anterior. Por essa razão, a ela será atribuído um peso maior, aumentando a probabilidade de que ocorra. Dessa forma, para o vetor aleatório A gerado, o vetor X será definido de acordo com a Figura 43.

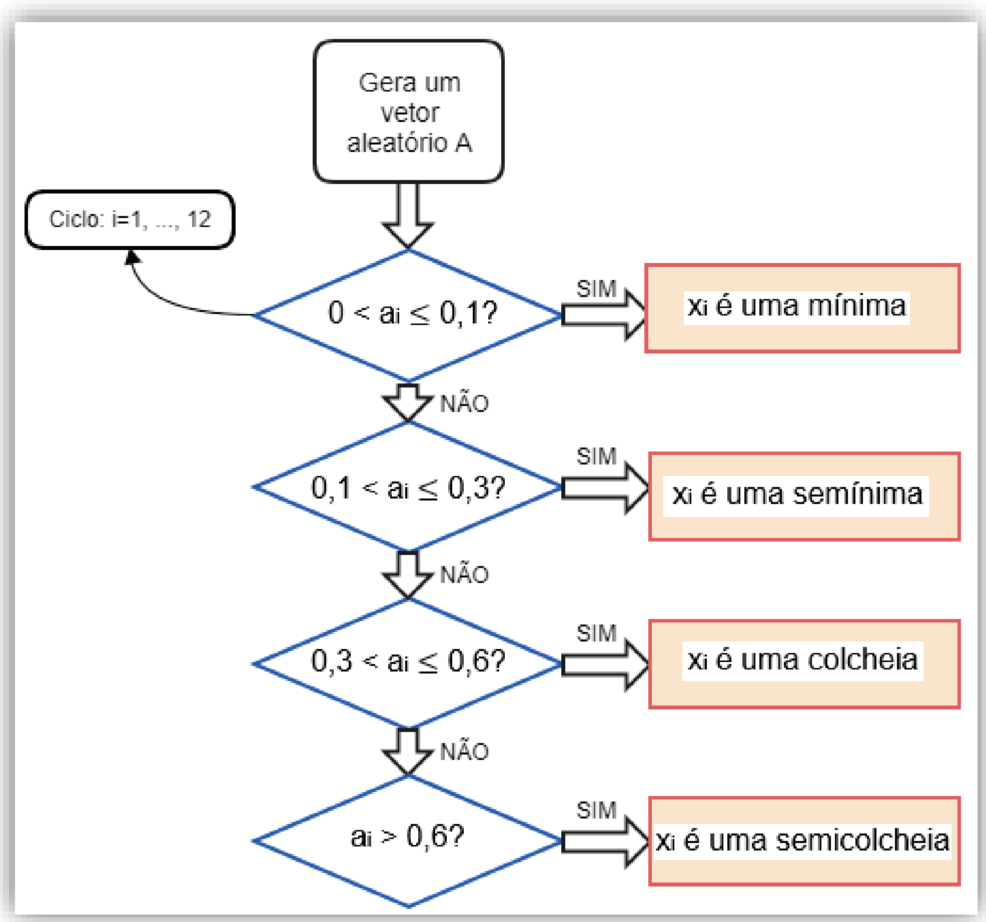

Figura 43: População Inicial – Algoritmo 6

De maneira análoga ocorre ao indivíduo Y gerado pelo vetor B, e uma vez formada a população inicial é necessário distribuir as figuras rítmicas entre os compassos, para que se possa calcular as funções fitness. Para isso, deve-se considerar o valor de cada figura de acordo com o compasso ao qual ela pertence. Ao observar a Figura 42, nota-se que há uma relação entre os valores do primeiro e do segundo esquema, de modo que uma figura rítmica em um compasso com denominador 4 vale o dobro que a mesma figura em um compasso com denominador 2.

Logo, ao definir a população inicial X e Y, preenche-se cada uma das entradas dos dois vetores com os valores da parte superior da Figura 42. Por exemplo, se o aleatório a_i gerado é igual a 0,23, a figura rítmica será a semínima (como mostra o esquema da Figura 43), e o valor atribuído a essa entrada será igual a 1. No entanto, se no decorrer da distribuição essa semínima acabar pertencendo ao terceiro ou quarto compasso, seu valor deverá ser igual a 0,5. Logo, para uma figura rítmica qualquer, basta definir seu valor baseando-se no denominador 4 e dividi-lo por 2 caso pertença ao terceiro ou ao quarto compasso.

Sendo $u_{(2,\ k)}$ o último elemento do segundo compasso, basta que a partir dessa posição todas as entradas passem a ter metade do valor original, e esse comando ao ser aplicado aos vetores iniciais garante a correta criação dos compassos.

Essa é principal diferença entre o programa 5 e este, já que a ideia geral permanece a mesma e uma vez atribuídos corretamente os valores às posições do vetor o que segue é semelhante. Logo, segue-se calculando as funções fitness, definem-se os pontos de cruzamento, geram-se os filhos e sobre eles analisam-se também as funções objetivo. Com as funções fitness dos pais e dos filhos calculadas, selecionam-se os dois melhores adaptados e repete-se o ciclo, até que uma solução ótima do problema seja obtida ou então até que a solução fique estagnada por iterações consecutivas, não obtendo melhora. Se isso ocorrer, aplica-se a mutação e o algoritmo é encerrado, da mesma forma que antes.

Ao aplicar a mutação outra diferença surge, e ela diz respeito novamente ao critério de escolha das figuras, que ocorre como mostra a Figura 44, onde a_k é o aleatório gerado e x_k é a posição que sofre a mutação, sendo k=i, j.

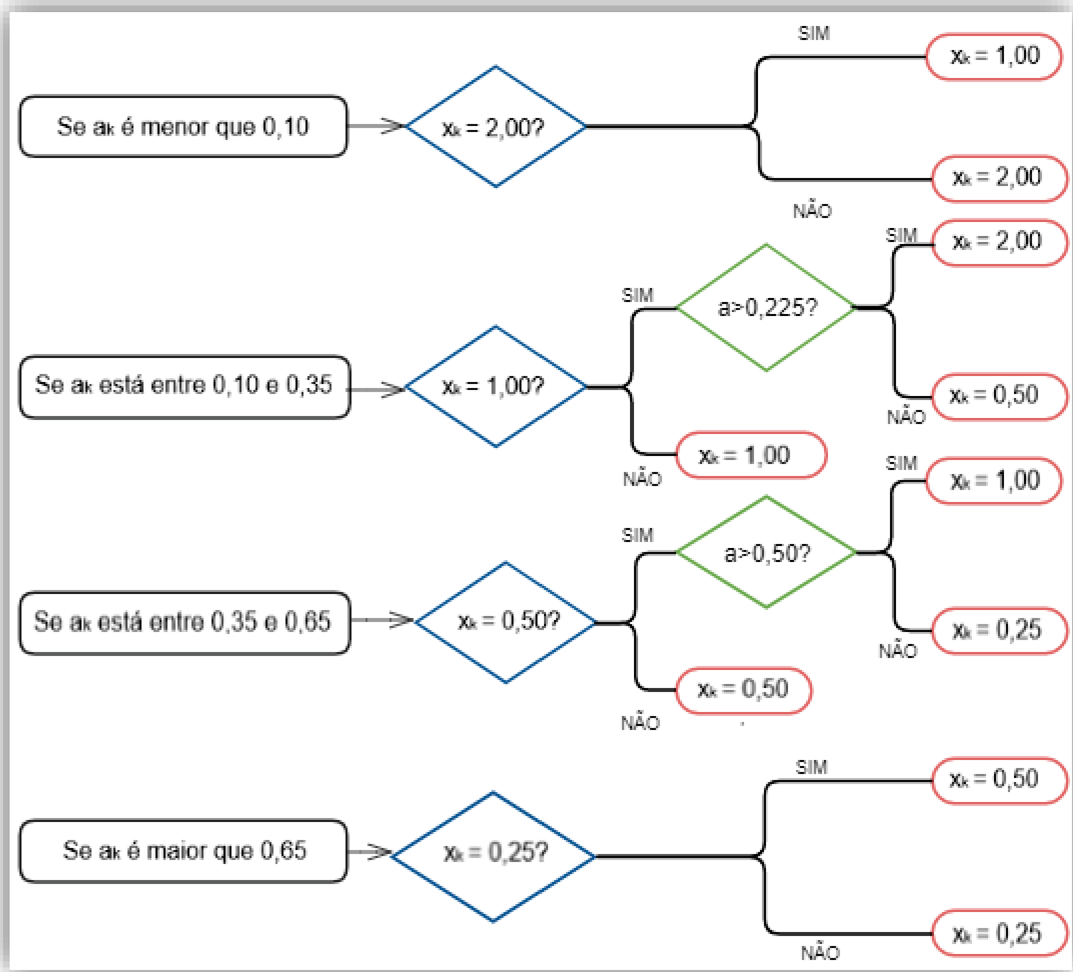

Figura 44: Mutação - Algoritmo 6

Sendo assim definido o critério de escolha da figura rítmica a partir do aleatório gerado, a probabilidade que a mutação gere uma mínima é menor que a probabilidade de uma semínima, e assim por diante, pois novamente foram atribuídos pesos distintos às figuras.

Selecionando os indivíduos melhores adaptados a partir das funções fitness dos compassos, o resultado obtido pode ser acompanhado a seguir.

Resultados: A Figura 45 apresenta o resultado obtido, tanto em forma vetorial como na partitura.

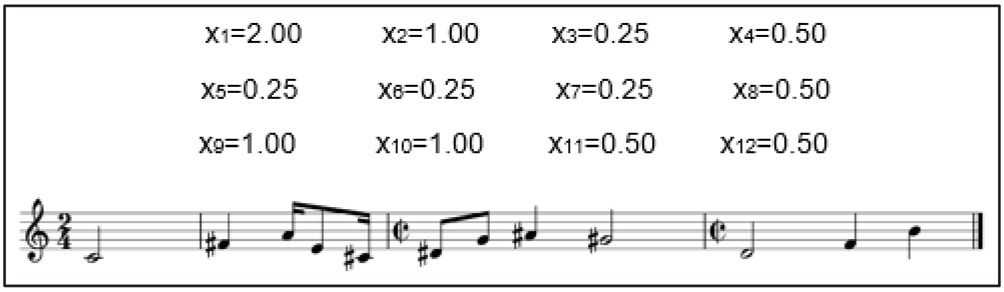

Figura 45: Solução ótima obtida através do algoritmo 6

Note que o valor 1 que ocupa a posição x_2 não é o mesmo que o valor 1 que ocupa a posição x_9, isso porque o primeiro pertence a um compasso 2 por 4, enquanto o outro pertence a um compasso 2 por 2. Logo, ao primeiro associa-se uma semínima, enquanto que ao segundo uma mínima, e com isso encerram-se as aplicações propostas.

5 CONSIDERAÇÕES FINAIS

A partir do que foi exposto e fazendo uma análise das soluções obtidas, verifica-se que foi possível chegar às soluções ótimas dos problemas propostos, gerando em cada um deles compassos formados por uma série dodecafônica, a partir das figuras rítmicas utilizadas.

Isso foi possível devido à relação criada entre os objetos musicais e os elementos característicos do PCV, PM e PMC, já que ao associar as notas musicais às cidades, as figuras rítmicas aos pesos e o compasso à mochila, bastou adequar as funções objetivo para que o algoritmo pudesse ser desenvolvido e retornasse os valores obtidos. Além disso, em cada caso a solução obtida pôde ser representada na forma de partitura, o que demonstra de fato que o objetivo real foi alcançado.

Ainda que aqui não se tenha extendido os resultados, aplicando as ideias discutidas anteriormente de inversão e retrogradação, é fácil perceber que após simples manipulações da série gerada, várias outras – decorrentes da primeira – podem ser obtidas, a partir das transformações isomórficas apresentadas. Então, embora tais combinações não tenham sido analisadas, fato é que sabe-se ser possível gerar uma composição dodecafônica a partir do que foi obtido. Essa etapa, no entanto, fica a critério do compositor, pois como foi citado logo no início o objetivo aqui é apenas o de gerar sequências e formar compassos, restando sempre a liberdade do criador como peça final na manipulação desses resultados.

O trabalho todo foi concentrado em aplicar os resultados obtidos pelo primeiro algoritmo, obtendo dessa forma soluções que giravam sempre em torno das mesmas notas. Nada impede, no entanto, que isso se expanda para outras sequências, que como sabemos, possuem inúmeras possibilidades combinatórias de acontecer.

Conclui-se reafirmando que em hipótese alguma deseja-se substituir o papel do compositor no processo de criação musical, e o que aqui foi desenvolvido visa apenas aplicar métodos heurísticos e metaheurísticos num campo pouco discutido e explorado, que é o da música.

5.1 RECOMENDAÇÕES PARA TRABALHOS FUTUROS

Mesmo tendo alcançado os objetivos, para aprimorar a variedade dos resultados obtidos nesse trabalho recomenda-se que projetos futuros possam desenvolver melhores meios de obter os números aleatórios necessários ao processo, e quem sabe alterar o software de programação utilizado, a fim de que se possa comparar os resultados.

Mas mais importante que isso, sugere-se a criação de algoritmos que efetuem as inversões e transposições da série, combinando-os e testando as composições que daí resultarem.

REFERÊNCIAS

MARQUES, F. do. P; ARENALES, M. N. **O problema da mochila compartimentada e aplicações**. Departamento de Ciências de Computação e Estatística, Universidade de São Paulo, São Carlos – SP, 2002.

CUNHA, C. B; BONASSER, U. de. O; ABRAHÃO, F. T. M. **Experimentos computacionais com heurísticas de melhoria para o problema do caixeiro viajante**. Trabalho apresentado no XVI Congresso da Anpet – Associação Nacional de Pesquisa e Ensino em Transportes.

SOARES, G. L. **Algoritmos Genéticos: Estudo, Novas Técnicas e Aplicações**. UFMG, Belo Horizonte, 1997.

VITOR, A. **Uma proposta de algoritmo genético híbrido para o problema do caixeiro viajante**. UFPR, Curitiba, 2015.

BENEVIDES, P. F. **Aplicação de heurísticas e metaheurísticas para o problema do caixeiro viajante em um problema real de roteirização de veículos**. UFPR, Curitiba, 2011.

SUCUPIRA, I. R. **Métodos heurísticos genéticos: Meta-heurísticas e Hiper-heurísticas**. USP, São Paulo, 2004.

SIQUEIRA, P. H. **Metaheurísticas e aplicações**. UFPR, Curitiba, 2016.

CORRÊA, A. F. **Estruturações Harmônicas Posteriores à Prática Comum – pantonalidade**. São Paulo, 2004. Dissertação de Mestrado, Universidade Estadual Paulista – Unesp.

DALLIN, L. **Techniques of Twentieth Century Composition**. 3ª edição. Iowa: WM. C. Brown Company Publishers, 1975.

KOSTKA, S; PAYNE, D. **Tonal Harmony – with an Introduction to Twentieth-Century Music**. 4ª ed. Boston: Mcgraw-Hill Inc., 2000.

SANTOS, A. C. dos. **Composição musical: estudos dirigidos para audiovisual**. UFMT, Instituto de Linguagens, Cuiabá, 2008.

MÚSICA, Descomplicando a. **Notas Musicais**. Disponível em: http://www.descomplicandoamusica.com/notas-musicais/. Acesso em: 01 julho de 2017.

POZO, A. *et al.* **Computação evolutiva**. Grupo de Pesquisas em Computação Evolutiva, Departamento de Informática, UFPR, 2011.

HOLLAND, J. H. **Adaptation in natural and artificial systems**. The University of Michigan Press, Ann Arbor, MI, 1975.

MITCHELL, M. **An introduction to genetic algorithms**. Cambridge: Mit Press. 1997. 207 p.

CUNHA, C. B. da; BONASSER, U. de. O; ABRAHÃO, F. T. M. **Experimentos computacionais com heurísticas de melhorias para o problema do caixeiro viajante**. Escola Politécnica da Universidade de São Paulo, 2002.

CARVALHO, R. **Problema da mochila**. Universidade Estadual de Campinas Instituto de Matemática, Estatística e Computação Cientifica – SP, 2015.

FREIRE, R. D. **Características e focos de aprendizagem de diversos sistemas de solfejo**. Departamento de Música da Universidade de Brasília, 2005.

HOLLAND, J. H. **Adaptation in natural and artificial systems.** The University of Michigan Press, Ann Arbor, MI, 1975.

SCHWEER, W. **MUSESCORE 2**, version 2.0.3. Publicado sob a GNU General Public License.

MICROSOFT, **Fortran PowerStation**, version 4.0. Microsoft Developer Studio, 1995.