Tapete Musical de Sachs

Thiago Amendola

Ciência da Computação - Graduação
E-mail: t148062@students.ic.unicamp.br
tico.amendola1@gmail.com

Resumo – Neste trabalho, foi proposto um algoritmo genético para encontrar uma permutação que maximize o encaixe de peças em um protótipo de tapete musical. O tapete em questão é composto por peças hexagonais cujos encaixes de cada face são representados por feixes de bits pareados de valores equivalentes. Além disso, as peças podem ser customizadas, de modo a rotacionar, inverter bits ou espelhar seu respectivo feixe de bits. O algoritmo proposto leva em conta a disposição das peças e suas respectivas configurações para gerar uma solução que maximize a quantidade de bits de encaixe por meio da criação de gerações de soluções e do uso de crossovers e mutações.

Palavras-chave: algoritmos evolutivos, algoritmos genéticos

1. Introdução

Um músico propôs a elaboração de um tapete musical de propósito lúdico e educativo para ser usado por crianças. Este tapete seria composto por diversas peças hexagonais configuráveis, onde cada uma emite um som diferente ao serem pisadas. Essas peças se encaixam como um quebra-cabeça, de maneira similar aos pedaços de um tatame, de modo a tornarem sua montagem e configurações de construção altamente customizáveis e escaláveis a cada tipo de situação. O intuito principal deste aparato é de usá-lo para o estudo de rítmica para crianças, desenvolvendo noções musicais e lógicas através da interação com o tapete e a reconstrução e reposicionamento do mesmo.

Do ponto de vista tecnológico, as peças do tapete seriam compostas primariamente por um feixe de 32 bits que se estenderia pelas bordas de seu formato hexagonal. Estes bits se acomodam a cada face do hexágono assim como mostrado na **Figura 1**. Alguns bits do feixe são atribuídos aos vértices do hexágono e contam para ambas as faces que geram o respectivo vértice. Cada face conta com 6 bits no total, exceto as faces superior e inferior, que contam com 7 bits ao todo.

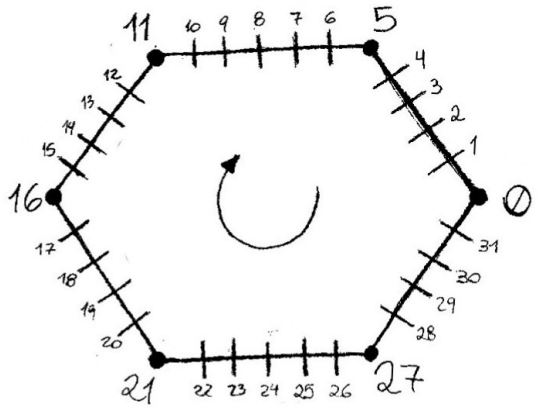

***Figura 1**: Representação da peça hexagonal. Os números presentes nas bordas indicam a distribuição de bits em cada face. Os bits de vértice contam para ambos os lados que o geram.*

Os bits presentes em uma face definem o encaixe da mesma. Isso significa que, para encaixar uma peça em cima de outra, a face de baixo da peça de cima deve conter os mesmos valores de bits que os da face de cima da peça de baixo. O mesmo se aplica às outras faces. Além disso, as peças possuem alguns parâmetros de configuração individual: a rotação de bits, a inversão dos bits e o espelhamento do feixe. Com o intuito de encontrar uma configuração de tapete cuja quantidade de encaixes, ou seja, de bits iguais entre faces adjacentes seja máxima, foi desenvolvido um algoritmo genético capaz de encontrar soluções boas e/ou ótimas. Consideraremos para a solução final um tapete de 32 peças por 16 peças, totalizando 512 peças. Além disso, consideraremos que o tapete respeita uma topologia toroidal, isto é, a peça do topo do tapete deve se encaixar com a respectiva peça do fundo do tapete, sendo a mesma regra aplicada às laterais.

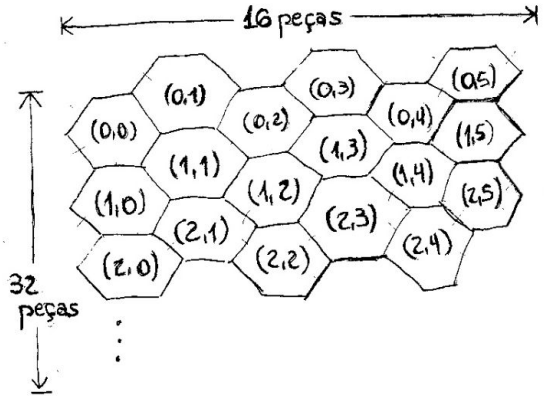

***Figura 2:** Representação da matriz de peças*

2. Trabalho Proposto

A representação do tapete foi feita por meio de uma matriz de elementos hexagonais (**Figura 2**). Para detectar as peças vizinhas a uma determinada peça, é avaliada a divisibilidade do valor da coluna por 2, visto que peças de coluna ímpar são mais elevadas e se relacionam de maneira diferente com seus vizinhos. Além disso, o tapete respeita uma topologia toroidal, o que significa que as peças de uma determinada borda se encaixam com as da borda diametralmente oposta. Isso não só garante que todas as faces do tapete sejam consideradas no cálculo do encaixe de bits, como permite a representação de soluções iguais dispostas em permutações diferentes.

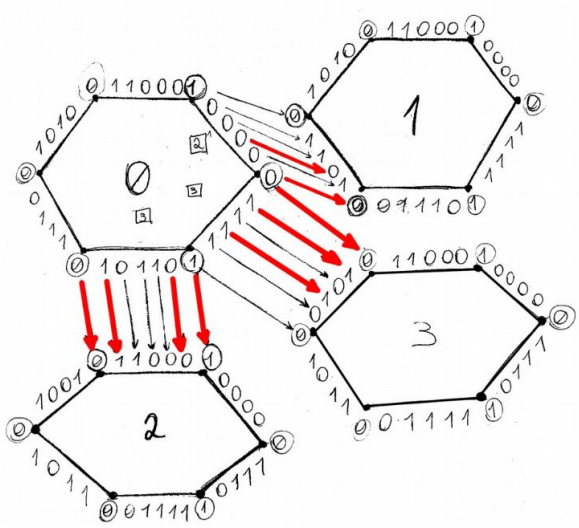

Figura 3: *Exemplo de checagem de pareamento para os lados superior-direito, inferior-direito e inferior da peça 1. As setas vermelhas representam os bits pareados de valores iguais*

Cada peça ainda contém três parâmetros individuais capazes de customizá-la: a rotação de bits, a inversão dos bits e o espelhamento da peça. O primeiro parâmetro indica em quantos bits o feixe foi rotacionado, considerando esta rotação no sentido horário. O segundo parâmetro é a inversão de bits, isto é, caso seja ativado, todos os valores de bits iguais a 0 se tornam 1 e vice versa. O último parâmetro representa o espelhamento da peça. Se verdadeiro, o feixe encontra-se espelhado. Será utilizado para o algoritmo genético desenvolvido uma função de fitness equivalente a quantidade de bits de encaixe equivalentes, ou seja, a quantidade de bits pareados de valor equivalente em cada um dos encaixes entre peças. É possível encontrar o valor de fitness checando, para cada peça, três de suas faces não opostas uniformemente por todo o tapete (**Figura 3**). Pela propriedade da topologia toroidal, uma peça cuja face testada esteja na borda do tapete terá a respectiva face pareada com a correspondente face da peça da outra borda do tapete. Foram consideradas nesta solução as faces superior-direita, inferior-direita e inferior. Para a peça 1 representada no exemplo da **Figura 3**, somando todos

os bits pareados obtemos um valor de fitness igual a 9. Ao repetir o processo para todas as peças, a soma obtida equivale ao valor de fitness daquela resposta.

De modo a formar um feixe cíclico capaz de apresentar todos os valores possíveis para os lados do hexaedro com maior lado de 7 bits, utilizamos um vetor pré-fornecido com sequências de DeBrujin, com alfabeto contendo 0 e 1 apenas e com tamanho de subsequência equivalente a 7. O vetor fornece 2048 sequências de feixe, sendo 512 sequências distintas repetidas 4 vezes.

3. Metodologia

Para a resolução deste problema, foi desenvolvido um algoritmo genético que capta um conjunto inicial de peças disponíveis. Para a execução do algoritmo deste trabalho, foi utilizada uma entrada com 2048 peças, contendo 512 peças distintas. Em seguida o algoritmo gera uma população inicial de soluções aleatórias, baseadas na permutação das peças fornecidas, na matriz correspondente ao tapete e nas diferentes configurações de parâmetros de cada peça. Construída a população inicial, são obtidos os valores de fitness para cada uma das soluções pertencentes a dada população, que é então ordenada seguindo este valor.

Desta população são selecionados pares, segundo um dado critério, para a realização de um processo de crossover através do método de *Partially Matched Crossover (PMX)*, gerando um par de novas soluções. A seleção dos pares é feita obtendo uma solução aleatória dentro da metade das melhores soluções e uma solução aleatória dentro da metade das piores soluções, segundo o parâmetro de fitness. O número de novas soluções a ser gerado durante cada iteração é fornecido previamente.

O nova população de soluções pode ser submetidas, segundo dada probabilidade, a um processo de mutação, que pode alterar as configurações de algumas das peças. A mutação de uma solução pode ocorrer no feixe de uma peça, na disposição de parâmetros de uma peça ou na própria disposição do tapete, onde cada uma das situações de mutação possui um hiperparâmetro pré-estipulado que define a probabilidade de tal evento.

O processo de encontrar a fitness, gerar novas soluções via crossover e mutá-las é repetido por um número finito de 20 gerações. De modo a não ficar preso ao problema da súbida da encosta, são feitas várias populações distintas entre si, onde o valor de fitness final é o melhor resultado dentre estas populações.

A linguagem utilizada foi C, visto que é uma linguagem versátil para a construção de estruturas de dados diversificadas e por ser uma das linguagens com maior rapidez em desempenho, fator essencial para um problema que exige grande esforço computacional.

4. Resultados

Através da implementação obtida, os valores para fitness máximo obtido variam entre 5153 e 5194. Como a solução ótima para o problema apresenta o fitness máximo 9728, só foi possível encontrar soluções com fitness de aproximadamente metade do valor máximo possível. É possível apontar através deste experimento que pode não existir uma solução perfeita para o problema proposto, ou seja, que seja capaz de alcançar o valor máximo de fitness. Também é possível apontar que melhorias no método de crossover e na estipulação do hiperparâmetros de crossover e mutação podem levar a soluções melhores.

5. Conclusão

A utilização de práticas de algoritmos genéticos possibilitou uma visão preliminar da complexidade existente no problema do tapete de Sachs. Embora seja provável que não exista uma solução ótima, as implementações possibilitaram um ponto de vista empírico quanto à distância entre a solução ótima hipotética e as soluções alcançáveis.

Reabordando o problema do tapete como uma solução lúdica musical para ensino infantil, é possível apontar que a simplificação da estrutura tecnológica proposta não só facilitaria a viabilidade da solução, como facilitaria o uso e o entendimento de seu funcionamento por parte de quem usará o equipamento. A diminuição do tamanho do feixe, do número de bits por lado da peça ou até mesmo a simplificação da interação entre peças encaixadas ajudaria com a facilitação e viabilidade do problema.

Referências

[1] Columbini, Esther. Introdução a Inteligência Artificial – Computação Evolutiva parte I. Acessado em 29/04/16: http://www.ic.unicamp.br/~esther/teaching/2016s1/mc906/Aula9_s.pdf

[2] Levine, Lionel. De Bruijn Sequences. Acessado em 14/05/16 em: http://www.math.cornell.edu/~levine/18.312/alg-comb-lecture-21.pdf

[3] Lacerda, Estéfane G. M. de. O problema do Caixeiro Viajante. Acessado em 14/05/16 em: http://www.dca.ufrn.br/~estefane/metaheuristicas/ag_pcv.pdf