Seat Assignment e Algoritmos Genéticos

Marcelo A. Teixeira *
*Ciência da Computação - Graduação
E-mail: marceloat01@gmail.com

***Resumo*—**
***Palavras-chave*—Seat Assignment Algoritmos Genéticos**

I. Introdução

O problema de Seat Assignment se baseia em decidir em quais mesas pessoas irão se sentar, tendo em vista a relação umas com as outras, descrita em uma tabela. Essa tabela é dada em forma de numeros, de forma que quanto maior for o numero melhor é a relação entre as pessoas. A cada mesa pode ser dado um valor que é a soma das relações entre todas as pessoas sentadas ali, queremos otimizar esse valor.

Removendo as restrições, esse problema pode ser resovlido por algortimo bruteforce[2], com tempo fatorial. Então utilizei algoritmos genéticos para resolver o problema em tempo consideravelmente melhor.

Foram utilizados duas versões do algortimo nesse trabalho, em cada uma modificando alguns métodos, e nos dois foram variados todos os parametros para fins de comparação.

II. Linguagem de programação

A linguagem escolhida nesse trabalho foi o *Javascript* por dois motivos: o autor possui amplo conhecimento e prática com a linguagem; e unindo as tecnologias web de HTML, CSS, entre outros, é possivel demonstrar resultados em uma pagina de browser com gráficos e opçõpes dinamicas de parametros.

O código fonte submetido com esse trabalho engloba todos os arquivos necessarios e um README com as instruções de como visuzalizar os resultados.

A. Modelagem

O problema foi moldado na forma de objetos (ou protótipos, do javascript), uma *População* possui um conjunto de *salões*, por sua vez, cada salão possui um conjunto de *mesas*, e cada mesa possui seus *ocupantes*. As palavras grifadas são os protótipos e possuem métodos para manipular seus atributos.

O valor de uma mesa é calculado somando as relações de todos os seus ocupantes segundo uma tabela dada. O valor do salão é a soma de todas as suas mesas, e esse é o valor que queremos otimizar no final de todas as iterações.

A tabela de entrada foi criada aleatoriamente, com 100 pessoas, e foram consideradas 10 mesas com 10 ocupantes cada.

III. Trabalho Proposto

Foram feitos dois algoritmos diferentes, cada um contendo métodos distintos de certos parametros, descritos abaixo:

Algoritmo 1:

substituição Troca-se todos os individuos da população pai pela população filho, exceto pelo mais apto (Elitismo).

seleção Torneio, escolhe-se um numero X de individuos aleatorios da população e é retornado o mais apto.

critério de parada Numero de gerações, foi escolhido o numero 1000.

mutação swap, uma quantidade aleatoria de individuos de duas mesas escolhidas aleatoriamente são trocados de lugar

crossover ponto unico, o individuo filho recebe de 0 até X mesas de um pai, e de X até 10 mesas de outro.

Algoritmo 2:

substituição São gerados M individuos, eles substituem os M piores individuos da geração anterior.

seleção Método da Roleta, a população mantem um vetor de soma das aptidões, e um valor aleatorio é escolhido correspondete a um indice nesse vetor.

critério de parada O algoritmo para quando não houver melhora no resultado nas ultimas P iterações

mutação swap de sequencia, uma mesa inteira tem seus individuos trocados com outras mesas, aleatoriamente.

crossover ponto duplo, o individuo filho recebe genes não sequenciais (com dois cortes) de dois individuos pais.

Tabela I
Algoritmo 1 - variando mutação

	Melhor resultado: 30269			
Pop	Torneio	Mutação	Média	Melhor
50	10	15%	28012,2	28544
50	10	30%	28262,6	29005
50	10	45%	28565,8	29458
50	10	60%	28921,2	29550
50	10	75%	29289,6	29727
50	10	90%	29933,6	30269

IV. Materiais e Métodos

V. Resultados e Discussão

VI. Conclusões

+————-+

Tabela II
ALGORITMO 1 - VARIANDO TORNEIO

	Melhor resultado: 29947			
Pop	Torneio	Mutação	Média	Melhor
50	10	75%	29289	29947
50	20	75%	28636	29136
50	30	75%	27975,6	28942
50	40	75%	28148,8	28802

Tabela III
ALGORITMO 1 - VARIANDO POPULAÇÃO

	Melhor resultado: 30024			
Pop	Torneio	Mutação	Média	Melhor
20	10	75%	28642	29104
40	10	75%	29579	29861
60	10	75%	29504	29637
80	10	75%	29695	30024
10	10	75%	29435	29657

[1] S. Gotshall, B. Rylander (1992). Optimal population size and the Genetic Algorithm. *School of Engineering, University of Portland*

[2] M. Bellows (2012). http://www.neatorama.com/2012/09/18/Finding-an-Optimal-Seating-Chart/. *Princeton University*