Задание 2. Генетические алгоритмы

Студеникина Мария Александровна
studenikina.marya@yandex.ru

Ниже приведены графики зависимостей средней и наименьшей ошибки от номера итерации для ферической функция, функции Розенброка и функции Растригина. Размер популяции 40, размер одного члена популяции 20.

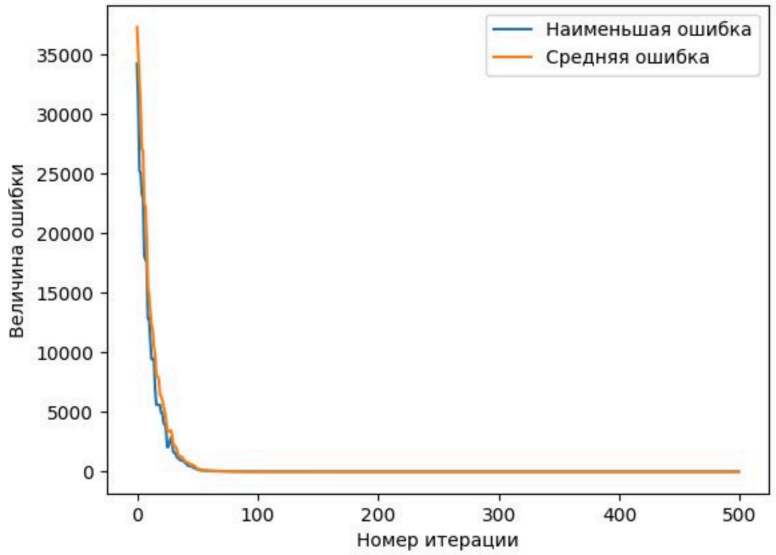

Рисунок 1. График для сферической функции

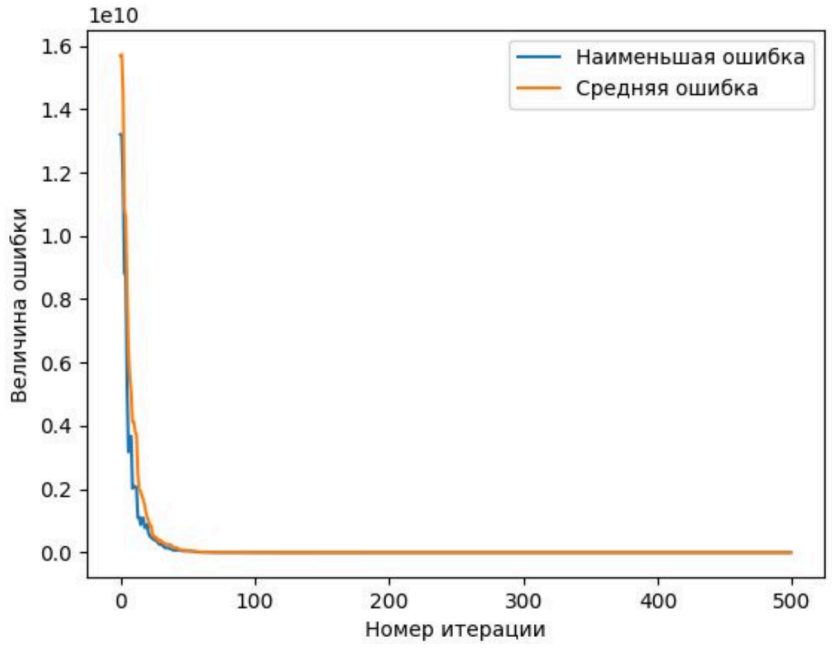

Рисунок 2. График для функции Розенброка

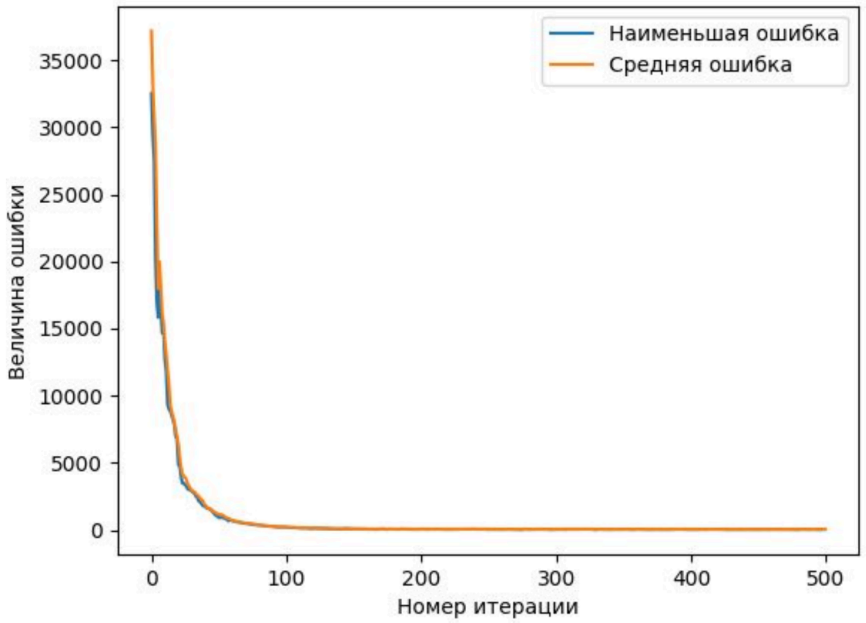

Рисунок 3. График для функции Растригина

Код

```
#include <iostream>
#include <cmath>
#include <fstream>
#include <mpi.h>

using namespace std;

double frand() // вещественное случайное число в диапазоне [0,1)
{
	return double(rand())/RAND_MAX;
}

int eval(int* a, int n)
{
	int sum = 0;
	for( int i=0; i<n; i++ )
	    sum += a[i];
	return sum;
}

void init(int* P, int m, int n)
{
	for( int k=0; k<m; k++ )
	    for( int i=0; i<n; i++ )
	        P[k*n+i] = rand()%2;
```

```
}

void shuffle(int* P, int m, int n)
{
	for( int k=0; k<m; k++ )
	{
		int l = rand()%m;
		for( int i=0; i<n; i++ )
			swap(P[k*n+i],P[l*n+i]);
	}
}

void select(int* P, int m, int n)
{
	double pwin = 0.75;
	shuffle(P, m, n);
	for( int k=0; k<m/2; k++ )
	{
		int a = 2*k;
		int b = 2*k+1;
		int fa = eval(P+a*n, n);
		int fb = eval(P+b*n, n);
		double p = frand();
		if( fa<fb && p<pwin || fa>fb && p>pwin )
			for( int i=0; i<n; i++ )
				P[b*n+i] = P[a*n+i];
		else
			for( int i=0; i<n; i++ )
				P[a*n+i] = P[b*n+i];
	}
}

void crossover(int* P, int m, int n)
{
	shuffle(P, m, n);
	for( int k=0; k<m/2; k++ )
	{
		int a = 2*k;
		int b = 2*k+1;
		int j = rand()%n;
		for( int i=j; i<n; i++ )
			swap(P[a*n+i],P[b*n+i]);
	}
}
```

```
void mutate(int* P, int m, int n)
{
	double pmut = 0.1;
	for( int k=0; k<m; k++ )
		for( int i=0; i<n; i++ )
			if( frand()<pmut )
				P[k*n+i] = 1-P[k*n+i];
}

double printthebest(int* P, int m, int n)
{
	int k0 = -1;
	int f0 = -1;
	for( int k=0; k<m; k++)
	{
		int f = eval(P+k*n, n);
		if( f>f0 )
		{
			f0 = f;
			k0 = k;
		}
	}
	//cout << f0 << ": ";
	//for( int i=0; i<n; i++ )
	// cout << P[k0*n+i];
	//cout << endl;
	return f0;
}

void runGA(int n, int m, int T, int rank, int size)
{
	int* P = new int[n*m];
	init(P, m, n);
	double best, sum;
	ofstream bFile("best2.txt");
	ofstream aFile("av2.txt");
	for( int t=0; t<T; t++ )
	{
		select(P, m, n);
		crossover(P, m, n);
		mutate(P, m, n);
		sum = best = printthebest(P, m, n);

		if (rank == 0) {
```

```
                MPI_Reduce(MPI_IN_PLACE, &sum, 1, MPI_DOUBLE, MPI_SUM, 0,
MPI_COMM_WORLD);
                MPI_Reduce(MPI_IN_PLACE, &best, 1, MPI_DOUBLE, MPI_MIN, 0,
MPI_COMM_WORLD);
                bFile << best << "\n";
                aFile << sum / size << "\n";
                cout << "It num - " << t << " best - " << best << " average - " <<
sum / size << "\n";
            } else {
                MPI_Reduce(&sum, 0, 1, MPI_DOUBLE, MPI_SUM, 0, MPI_COMM_WORLD);
                MPI_Reduce(&best, 0, 1, MPI_DOUBLE, MPI_MIN, 0, MPI_COMM_WORLD);
            }
        }

        delete[] P;
}

int main(int argc, char** argv)
{
        MPI_Init(&argc, &argv);
        int rank;
        int size;
        MPI_Comm_rank(MPI_COMM_WORLD, &rank) ;
        MPI_Comm_size(MPI_COMM_WORLD, &size) ;
        int n = atoi(argv[1]);
        int m = atoi(argv[2]);
        int T = atoi(argv[3]);

        runGA(n, m, T, rank, size);
        getchar();
        MPI_Finalize();
        return 0;
}
```