UNIVERSIDADE ESTADUAL DE MATO GROSSO DO SUL
CURSO DE CIÊNCIA DA COMPUTAÇÃO

Relatório sobre o desenvolvimento.:

Rastrigim Function e Radio Network Optimization

Trabalho acadêmico apresentado à disciplina de Programação Paralela e Distribuída, do curso de Bacharelado em Ciência da Computação como requisito parcial para obtenção da nota do terceiro bimestre.

Profª. Drª. Rubens Barbosa Filho

Felipe Lima Morais
Robson Takashi Kawakita
Rogers Prates de Pelle

Dourados - MS
Novembro de 2015

Sumário

1. Introdução

1.1. Algoritmos Genéticos

Algoritmos Genéticos são inspirados no princípio Darwiniano da evolução das espécies e na genética . São algoritmos probabilísticos que fornecem um mecanismo de busca paralela e adaptativa baseado no princípio de sobrevivência dos mais aptos e na reprodução (GOLBERG, 2015).

1.2. Rastrigim Function

O Rastrigim Function é um algoritmo utilizado para executar testes de otimização de algoritmos. A função Rastrigin tem vários mínimos locais, a localizações desses são distribuídos regularmente. A Figura 1 mostra a definição da função, e a Figura 2 mostra a representação gráfica na sua forma bidimensional.

$$f(\mathbf{x}) = 10d + \sum_{i=1}^{d} \left[x_i^2 - 10\cos(2\pi x_i) \right]$$

Figura 1: Difinição da Rastrigi Function

Fonte: http://www.sfu.ca/~ssurjano/rastr2.png

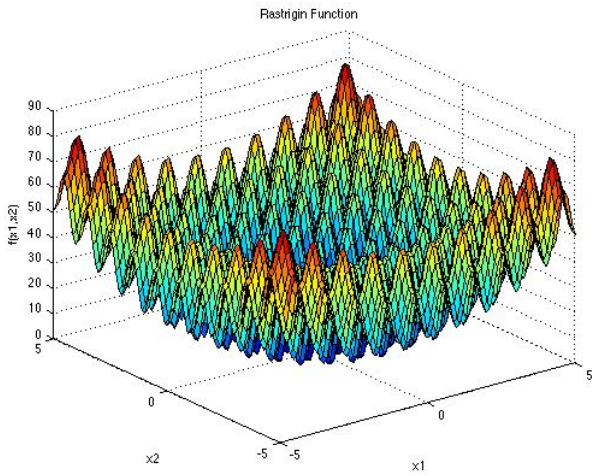

Figura 1: Representação gráfica da Rastrigi Function

Fonte: http://www.sfu.ca/~ssurjano/rastr.png

1.3. Radio Network Optimization

O Radio Network Optimization é um problema de otimização de cobertura de sinal. ele se propõe a descobri: dada uma determinada área, qual o melhor posicionamento das antenas afim de cobrir a maior área possível. Existem várias casos em que este problema pode ser aplicado, não apenas a distribuição de sinal. São considerados a área de cobertura de cada antena, os pontos onde elas se inspetam, o desperdício de cobertura fora da área programada entre outros. A Figura 3 expressa graficamente em 2D um exemplo de distribuição de antenas.

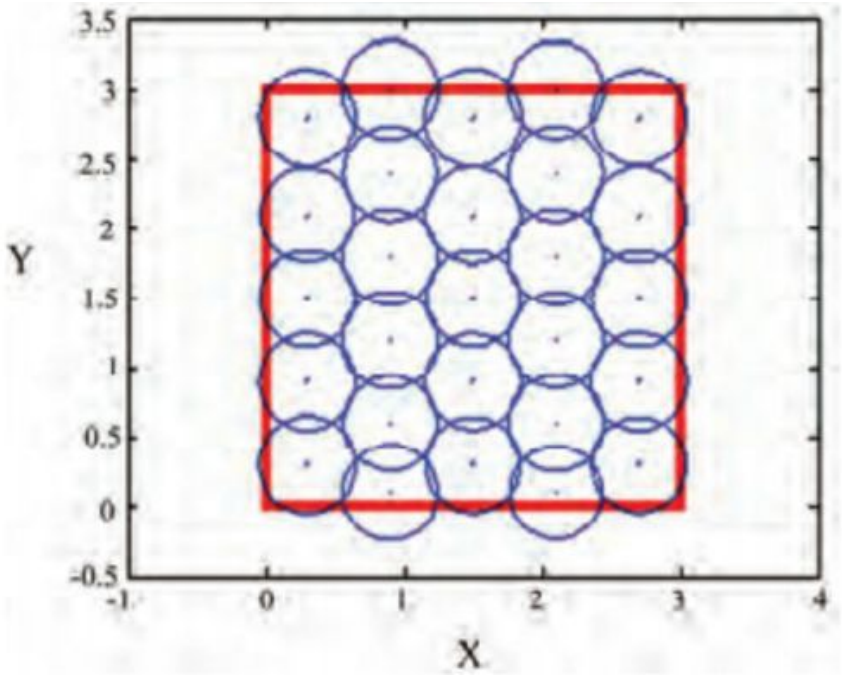

Figura 3: Exemplo de distribuição de antenas.

Fonte: http://www.sfu.ca/~ssurjano/rastr.png

1.4. Objetivos

O objetivo deste trabalho é implementar os algoritmos Rastrigim Function e Radio Network Optimization utilizando algoritmos genéticos e relatar seu desenvolvimento.

2. Implementação

2.1. Materiais

A linguagem *Python*, na versão 2.7, foi utilizada no desenvolvimento dos algoritmos de do *middleware*. A *Message Passing Interface* (MPI) foi utilizada para distribuir a execução entre as máquinas, e a biblioteca *mpi4py* na comunicação dos algoritmos e o MPI.

2.2. Middleware

Os algoritmos foram modelados como classes, compostas por 6 métodos cada que são utilizados pelo *middleware*:

- **new_individual()** - Retorna uma lista de '0' ou '1' aleatória como representação do novo individuo.
- **get_fitness()** - Verifica o quanto próximo esta da resposta, quanto mais perto, retorna uma resposta próxima de **zero**.
- **is_finished()** - Verifica se o indivíduo é uma resposta aceitável do problema do problema.
- **validate_individual()** - Verifica se um indivíduo é válido para o domínio do problema.
- **num_bits()** - Retorna o numero de bits de um individuo.
- **show(individual)** - Imprime um individuo na tela.

2.3. Rastrigim Function

O método new_individual() da classe Rastrigim gera um indivíduo composto por 30 *bits* que representam 3 números inteiros variando seu de 0 à 1024. O método validate_individual() verifica se os números gerados estão no intervalo de -512 a 512, que é o domínio da função. O get_fitness retorna a proximidade da resposta, conforme mostra a implementação na Figura 4.

```
def get_fitness(self, individual):
    v = []
    for i in xrange(AMOUNT_NUM):
        number = int(''.join(individual[i*NUM_BITS_IN_NUM:(i+1)*NUM_BITS_IN_NUM]), 2)
        v.append((number - 512)/100.0)

    alpha = 10
    fitness = 0
    for i in range(AMOUNT_NUM):
        fitness += v[i]**2 - alpha*math.cos(2*math.pi*v[i])
    return float(fitness) + alpha*AMOUNT_NUM
```

Figura 4: Método get_fitness da classe Rastringim

Fonte: Elaborada pelos autores

2.4. Radio Network Optimization

A classe RadioNetwork gera indivíduos compostos por uma sequência de coordenadas de antenas, estas são geradas em binário de forma aleatória e validadas se pertencem a área a ser coberta através do método validate_individual(), demonstrado na Figura 5. Todas as especificações de área, cobertura de antena, numero de anteras podem ser atribuídas nos atributos da classe. O get_fitness verifica o quanto da área foi coberta, levando em consideração a área de intersecção das antenas, subtraindo da área coberta. A sua implementação se encontra na figura 5.

```
def validate_individual(self, individual):
    """Validation of an individual to find out if it is part of the domain"""
    for i in xrange(self.amount_bs * 2):
        number = int(''.join(individual[i*self.NUM_BITS:(i+1)*self.NUM_BITS]), 2)
        if math.fabs(number) > self.limit_area - self.covered_bs or math.fabs(number) < self.covered_bs:
            return False, individual
    return True, individual
```

Figura 4: Método validate_individual da classe RadioNetwork

Fonte: Elaborada pelos autores

3. Resultados

3.1. Ambiente de Testes

O ambiente de testes foi Laboratório de Informática I, foram utilizados 5 computadores, suas especificações são descrita abaixo:

Processador	Intel i5-240 3.5GHz 6MB Cache
Memória RAM	4GB DDR3
Armazenamento	500GB
Sistema Operacional	Ubuntu 14.04 (64 bits)

3.2. Rastrigim Function

Modelos Probabilísticos Gerados

Teste	25%	50%	75%	100%
1º	2	507	901	901
2º	5	519	901	901
3º	4	492	900	901
4º	1	526	900	901
5º	3	517	898	901
6º	3	540	900	901

Tabela 1: Execução do Rastrim Function com uma população de 900 indivíduos

Fonte: Elaborada pelos autores

3.3. Radio Network Optimization

Modelos Probabilísticos Gerados

Teste	25%	50%	75%	100%
1º	1	406	882	901
2º	2	453	895	901
3º	1	430	886	901
4º	5	430	885	901
5º	1	441	891	901
6º	1	440	892	901

Tabela 1: Execução do Rastrim Function com uma população de 900 indivíduos

Fonte: Elaborada pelos autores

4. Referencias

GOLBERG, David E. Genetic algorithms in search, optimization, and machine learning. Addion wesley, 1989.

SURJANOVIC, Sonja; BINGHAM, Derek. Optimization Test Problems Rastingin Function. <http://www.sfu.ca/~ssurjano/rastr.html> Acessado em 18/11/2015