INSTITUTO POLITÉCNICO NACIONAL

Centro de Investigación en Computación

ASIGNATURA:

Metaheurísticas

Actividad #9:

Guía Taller Laboratorio 4

PROFESORA:

Dra. Yenny Villuendas Rey

PRESENTA:

Juan René Hernández Sánchez

Adriana Montserrat García Carrillo

1. Introducción

El término de búsqueda heurística viene del griego *heuriskein*, que significa encontrar. El enfoque heurístico intenta reducir el tamaño del árbol cortando nodos pocos prometedores. Además, está orientado a reducir la cantidad de búsqueda requerida para encontrar una solución.

El algoritmo heurístico “Ascensión de Colinas” toma su nombre de la semejanza que tiene con un alpinista, quien desea alcanzar rápidamente el pico de una montaña, este selecciona la dirección de ascenso mayor a partir de la posición actual. Por lo tanto, en este trabajo se estudió el algoritmo de Ascensión de Colinas con Mutación Aleatoria (RMHC), que es una variante del de Ascensión de Colinas; donde a partir de un estado, se genera otro mediante procesos aleatorios. Posee las siguientes características: no se inspeccionan todos los estados sucesores, sólo el generado, y si el estado generado supera al anterior, se considera como el estado actual.

Es importante recordar los ingredientes que conforman a los problemas de optimización, los cuales son:

- Función objetivo.
- Conjunto de parámetros (desconocidos) los cuales afectan el valor de la función objetivo. Se deben encontrar los parámetros que maximizan o minimizan la función objetivo.
- Conjunto de restricciones que restringen los valores que se pueden asignar.

En el presente trabajo se implementó el algoritmo de RMHC para la solución de problemas de minimización de distintas funciones.

2. Desarrollo

Asignatura: Metaheurísticas
Actividad No.9
Guía Taller No.4
Título: Solución de problemas mediante Ascensión de Colinas

Contenido:

▪ Métodos heurísticos de solución de problemas.
▪ Ascensión de Colinas con mutación aleatoria

Objetivo: Implementar algoritmos de Ascensión de Colinas, en lenguajes de alto nivel, para la solución de problemas de competencia.

1. Analice detalladamente las seis funciones definidas en el documento "Funciones de prueba.pdf". [1] [2]

Nombre de la función	Ref.	Fórmula	Punto mínimo	Valor mínimo
Alpine 1 Function	[1]	$f_1(x) = \sum_{i=1}^{D} \lvert x_i \sin(x_i) + 0.1x_i \rvert$	x* = f(0,0)	f(x*)=0
Dixon & Price Function	[2]	$f_2(x) = (x_1 - 1)^2 + \sum_{i=2}^{D} i(2\sin(x_i) - x_{i-1})^2$	x* = f(2($\frac{2^i-2}{2^i}$))	f(x*)=0
Quintic Function	[3]	$f_3(x) = \sum_{i=1}^{D} \lvert x_i^5 - 3x_i^4 + 4x_i^3 - 2x_i^2 - 10x_1 - 4 \rvert$	x* = f(-1 or 2)	f(x*)=0
Schwefel 2.23 Function	[4]	$f_4(x) = \sum_{i=1}^{D} x_i^{10}$	x* = f(0,0)	f(x*)=0
Streched V Sine Wave Function	[5]	$f_5(x) = \sum_{i=1}^{D-1} (x_{i+1}^2 + x_i^2)^{0.25} \left[sin^2 \left\{ 50(x_{i+1}^2 + x_i^2)^{0.1} \right\} + 0.1 \right]$	x* = f(0,0)	f(x*)=0
Sum Squares Function	[6]	$f_6(x) = \sum_{i=1}^{D} i x_i^2$	x* = f(0,0)	f(x*)=0

2. Implemente dichas funciones.

```python
#Alpine 1 Function
def fitness1(funcion):
    valmax = 0
    for i in funcion:
        valmax += abs(i*(math.sin(i))+0.1*(i))
    return valmax

#Dixon & Price Function
def fitness2(funcion):
    valmax = (funcion[0] - 1)**2
    for i in range(1, len(funcion)):
        valmax += i*(2*math.sin(funcion[i])-funcion[i-1])**2
    return valmax

#Quintic Function
def fitness3(funcion):
    valmax = 0
    for i in funcion:
        valmax += abs((i**5)-(3*(i**4))+(4*(i**3))-(2*(i**2))-(10*i)-4)
    return valmax

#Schwefel 2.23 Function
def fitness4(funcion):
    valmax = 0
    for i in funcion:
        valmax += i**10
    return valmax
```

```
#Streched V Sine Wave Function
def fitness5(funcion):
    valmax = 0
    for i in range(len(funcion)-1):
        valmax += ((funcion[i+1]**2 +
funcion[i]**2)**0.25)*((math.sin(50*((funcion[i+1]**2 +
funcion[i]**2)**0.1))**2)+0.1)
    return valmax

#Sum Squares Function
def fitness6(funcion):
    valmax = 0
    for j, xi in enumerate(funcion):
        valmax += j * (xi**2)
    return valmax
```

3. Implemente el algoritmo de RMHC para la solución de los problemas de minimización de las funciones anteriores. Considere D = 10 y posteriormente D = 30 dimensiones.

```
import random
import copy
import math

#Alpine 1 Function
def fitness1(funcion):
    valmax = 0
    for i in funcion:
        valmax += abs(i*(math.sin(i))+0.1*(i))
    return valmax

#Dixon & Price Function
def fitness2(funcion):
    valmax = (funcion[0] - 1)**2
    for i in range(1, len(funcion)):
        valmax += i*(2*math.sin(funcion[i])-funcion[i-1])**2
    return valmax

#Quintic Function
def fitness3(funcion):
    valmax = 0
    for i in funcion:
        valmax += abs((i**5)-(3*(i**4))+(4*(i**3))-(2*(i**2))-(10*i)-4)
    return valmax
```

```python
#Schwefel 2.23 Function
def fitness4(funcion):
    valmax = 0
    for i in funcion:
        valmax += i**10
    return valmax

#Streched V Sine Wave Function
def fitness5(funcion):
    valmax = 0
    for i in range(len(funcion)-1):
        valmax += ((funcion[i+1]**2 +
funcion[i]**2)**0.25)*((math.sin(50*((funcion[i+1]**2 +
funcion[i]**2)**0.1))**2)+0.1)
    return valmax

#Sum Squares Function
def fitness6(funcion):
    valmax = 0
    for j, xi in enumerate(funcion):
        valmax += j * (xi**2)
    return valmax

def mutar_bit(locus, funcion):
    new_funcion = funcion.copy()
    new_funcion[locus] = random.uniform(-10, 10)
    return new_funcion

#n representa a D, posteriormente n será igual 30
n = 10

#N representa el número de evaluaciones
N = 500

funcion_x = []
for i in range(n):
    funcion_x.append(random.uniform(-10, 10))

fitness_actual = fitness1(funcion_x)
eval = 1

while eval < N:
    locus = random.randint(0, n-1)
    nueva_funcion_x = mutar_bit(locus, funcion_x)
    nueva_fitness = fitness1(nueva_funcion_x)
```

```
    eval += 1

    if nueva_fitness <= fitness_actual:
        funcion_x = nueva_funcion_x
        fitness_actual = nueva_fitness

print("\n")
print("Valores Finales: ", funcion_x)
print("fitness igual a: ", fitness_actual)
print("\n")
```

4. Reporte los resultados obtenidos. Para ello, realice 20 ejecuciones independientes, con la siguiente configuración:

a. Considere un total de 500 evaluaciones de la función objetivo.

b. Muestre el mejor, peor, promedio, mediana y desviación estándar de los resultados en las 20 ejecuciones.

- ✓ **Resultados con D = 10**

Tabla de Fitness para D = 10

Función	Mejor	Peor	Promedio	Mediana	Desviación Estándar
F1	1.217596741	0.090578949	0.442647784	0.39293625	0.272357217
F2	55.6891453	1.90531574	20.32991462	16.12422627	13.98360807
F3	30.41182359	1.141753806	8.81557735	7.288663567	6.22072919
F4	0.103540541	3.10862E-06	0.012208823	0.000984463	0.024593258
F5	2.701923723	1.385216107	2.102710511	2.1042192	0.383728814
F6	13.64362453	0.580774099	3.280579006	2.756663509	2.851170743
Promedio	**17.29460907**	**0.850606968**	**5.830606348**	**4.777948877**	**3.956031215**

F1	F2	F3	F4	F5	F6
0.311908063	13.86267489	1.141753806	0.010991972	2.660095983	4.63010238
0.313027561	17.23671365	6.688586066	0.001503019	2.390978745	5.055183304
0.198021736	16.16426304	9.454237045	0.03527755	2.152513282	3.586614211
0.250388313	26.76838002	13.08133131	6.24E-05	1.93119822	3.780591087
0.443379284	32.94604634	11.01491885	8.49E-05	2.357724498	13.64362453
0.321238188	11.273227	11.37324606	0.000945893	2.103466889	3.09185913
0.686559251	28.48341101	9.571568386	0.002281304	2.578992628	1.291084313
0.49906489	43.06334367	6.765494202	0.020059782	1.849949497	1.177446912
0.337728625	8.302285074	30.41182359	0.000206563	1.765920304	4.715526872
0.463905286	8.34850302	1.876185166	0.001000449	1.741641184	3.138854236
0.729720008	2.51044738	7.811832931	0.000685696	2.104971511	3.231715727
0.424701626	16.0841895	3.942614344	3.11E-06	1.613472107	2.307982497

0.374073451	16.00682654	10.03136392	9.48E-06	1.385216107	1.337867729
0.133207557	27.68051311	6.148039894	4.44E-05	1.874831944	0.580774099
0.411799049	11.76223083	6.240830431	0.000968476	2.59236755	5.488831615
1.217596741	1.90531574	8.745093728	0.103540541	2.218875458	2.421467888
0.176205583	10.0249942	6.599294339	0.03223237	1.691792711	2.118740551
0.875160948	55.6891453	4.841421105	0.030315936	2.466115584	0.808892636
0.594690571	38.07149084	15.64871168	3.59E-06	2.701923723	1.086134278
0.090578949	20.41429115	4.923200137	0.003959015	1.872162301	2.118286127

- ✓ **Resultados con D = 30**

Tabla de Fitness para D = 30

Función	Mejor	Peor	Promedio	Mediana	Desviación Estándar
F1	10.36535183	3.663024179	5.798259936	5.224225174	1.75014681
F2	1076.813615	409.8393035	663.3497252	607.810802	213.2672347
F3	629.0793999	104.8850106	175.9581075	143.0852847	118.0529381
F4	100997.2236	4.245946077	14667.10952	1339.930787	29082.76813
F5	13.42461684	8.371325943	9.907679597	9.65272149	1.293332605
F6	514.7020229	124.1497342	277.1511467	256.4472523	115.1860281
Promedio	**17206.93476**	**109.1923908**	**2633.212407**	**393.6918454**	**4922.052968**

F1	F2	F3	F4	F5	F6
4.082396387	767.8065358	152.3299767	987.91154	9.361988815	303.1556962
4.489794443	608.9970216	307.5850174	923.2003342	10.09589179	181.5673494
7.135802542	433.484174	122.8309358	4.245946077	9.783867403	282.0984458
7.682311221	975.0719849	111.940133	5.40E+02	9.521575577	181.0264345
4.49429603	956.4593064	165.9612031	5.26E+02	9.195292782	188.5577308
5.346066986	661.1344524	115.1391069	58.51934874	13.42461684	180.5601739
5.102383361	483.5564267	247.1302144	17520.64836	10.64155963	215.3394231
6.523597729	995.6075506	136.1038178	386.0484087	8.422958027	514.7020229
3.663024179	835.3018587	161.5411024	45461.66721	8.399660024	487.1750903
6.189039175	499.5621709	116.5460726	1691.950033	8.371325943	318.6023938
7.279150947	606.6245824	104.8850106	19845.32186	8.864997056	124.1497342
4.497196901	484.3957561	111.7677996	2.57E+03	8.922200882	423.5690503
4.076033836	711.502803	111.5576591	9.52E+02	10.1259108	456.5355453
7.55382133	1076.813615	629.0793999	1.01E+05	11.93601738	214.5649323
7.23173216	443.6855416	148.0800351	85553.6679	11.39009248	268.9069302
4.767845246	509.9104419	111.2235225	23.43609311	10.46120633	190.3923514
4.061781593	603.3922561	184.2640223	880.6701982	9.518950867	243.9875744
10.36535183	774.4788839	189.5912076	4066.287078	8.841228322	151.6628567
7.060141887	409.8393035	153.515378	6.06E+03	10.91509667	343.1323258
4.363430935	429.3698384	138.0905343	4290.897846	9.959154327	273.3368729

c. Muestre el mejor, peor, promedio, mediana y desviación estándar de los tiempos de ejecución (en segundos) en las 20 ejecuciones.

- ✓ **Resultados con D = 10**

Tabla de Tiempos (segundos) para D = 10

Función	Mejor	Peor	Promedio	Mediana	Desviación Estándar
F1	0.01003838	0.0019877	0.00475526	0.004987	0.001905539
F2	0.01296353	0.0049839	0.00817882	0.00797832	0.002086418
F3	0.02393627	0.00997114	0.01495949	0.01296663	0.003870067
F4	0.00598478	0.00199485	0.00371472	0.00374329	0.000964291
F5	0.03290629	0.01196742	0.02175114	0.02243984	0.006170265
F6	0.00798035	0.0029912	0.00518614	0.00498819	0.00143281
Promedio	**0.01563493**	**0.00564937**	**0.00975759**	**0.00951721**	**0.002738232**

F1	F2	F3	F4	F5	F6
0.01003838	0.0049839	0.01296449	0.00299144	0.01196742	0.005982399
0.00570536	0.00698328	0.00997353	0.00299025	0.01296544	0.003989458
0.00598311	0.00698376	0.01296449	0.00398946	0.01496172	0.002992392
0.00299048	0.00697875	0.00997114	1.99E-03	0.01496363	0.004988909
0.00299191	0.00698161	0.01296735	2.99E-03	0.01714301	0.004987955
0.00698495	0.00499797	0.01196861	0.00399113	0.01695943	0.004978895
0.0039866	0.00896406	0.01196551	0.00398946	0.01894355	0.002993345
0.00498605	0.00598407	0.01296592	0.00299144	0.01695371	0.003989935
0.00299168	0.00797987	0.01097107	0.00297928	0.01695514	0.004986048
0.00498796	0.00698066	0.01195955	0.00399089	0.0199461	0.002991199
0.00398922	0.00898194	0.01296449	0.00299191	0.02593231	0.004988432
0.0019877	0.00697994	0.01496005	3.50E-03	0.02692747	0.004986763
0.00598574	0.00997305	0.01894951	2.99E-03	0.02493358	0.005982161
0.00598407	0.00797677	0.01595759	3.99E-03	0.02692771	0.00698328
0.00598264	0.00798011	0.0179522	0.00598478	0.02593136	0.006980896
0.00456786	0.00997281	0.01894999	0.00498509	0.03290629	0.00398922
0.0049901	0.0089767	0.01994705	0.00498843	0.0279243	0.00598526
0.00298786	0.00998759	0.02393627	0.00398803	0.02892733	0.005981207
0.00199485	0.01196599	0.01795125	4.99E-03	0.02592564	0.007980347
0.00498867	0.01296353	0.01894975	0.00299191	0.02692771	0.006984711

✓ **Resultados con D = 30**

Tabla de Tiempos (segundos) para D = 30

Función	Mejor	Peor	Promedio	Mediana	Desviación Estándar
F1	0.01097178	0.00399184	0.00745783	0.00698197	0.002017985
F2	0.01994729	0.00896955	0.01506054	0.01546001	0.003484548
F3	0.03490925	0.02094293	0.02812731	0.02842629	0.004417477
F4	0.02592897	0.00399327	0.01087084	0.00947857	0.004665474
F5	0.04886627	0.03190613	0.03816409	0.03556299	0.00596558
F6	0.02792311	0.00498462	0.01311722	0.01148164	0.005849878
Promedio	**0.02809111**	**0.01246472**	**0.01879964**	**0.01789858**	**0.004400157**

F1	F2	F3	F4	F5	F6
0.0049839	0.01096773	0.02094293	0.00399327	0.03522921	0.004984617
0.00636649	0.01196527	0.02397704	0.00798035	0.03889513	0.006979227
0.00601411	0.01795173	0.02194595	0.00897431	0.04587889	0.005022526
0.00399184	0.01795459	0.02793336	9.98E-03	0.04687428	0.011988163
0.00598812	0.01795173	0.02792788	1.10E-02	0.0458796	0.019932747
0.00602818	0.01895046	0.02792382	0.00897408	0.04886627	0.012965202
0.00599027	0.01795363	0.02891922	0.00897598	0.04687405	0.020942926
0.00697732	0.01994538	0.02992034	0.0089767	0.04089117	0.018948078
0.0050211	0.01994729	0.03091598	0.01495886	0.04288363	0.027923107
0.00598645	0.01695514	0.03091621	0.01296663	0.0379076	0.014962435
0.00897169	0.01695514	0.03291416	0.01895094	0.03589678	0.013960361
0.01001954	0.01796198	0.03390789	2.59E-02	0.03191447	0.01894927
0.00698662	0.00896955	0.03191447	7.97E-03	0.03191543	0.014966249
0.01097178	0.01296568	0.03091812	1.10E-02	0.03491735	0.007979155
0.00896811	0.01296616	0.03490925	0.01097131	0.03190613	0.010967731
0.0099721	0.0119679	0.03290749	0.0079782	0.03290939	0.010971785
0.00897598	0.01396489	0.02394056	0.00997329	0.03291059	0.009972811
0.00898194	0.01097083	0.02492881	0.00797582	0.03391075	0.010975122
0.00997591	0.01195359	0.02094483	1.10E-02	0.03291392	0.009976864
0.00798512	0.01199222	0.02393794	0.00898385	0.03390718	0.008975983

3. Conclusiones

Al implementar el algoritmo RMHC para la solución de problemas de minimización de distintas funciones se pudo apreciar que, es un algoritmo de gran versatilidad y es posible adaptarlo a casi cualquier problema de optimización.

Además, al obtener los resultados se pudo observar que la función 4 (Schwefel 2.23 Function) fue la que obtuvo el mejor desempeño, tanto en los resultados de la función fitness como en su tiempo de ejecución cuando D fue igual a 10. Por otro lado, la función 1 (Alpine

1 Function) fue la que obtuvo los mejores resultados en tiempo de ejecución y en la función fitness cuando D fue igual a 30.

4. Referencias

[1] Mitchell, M., Holland, J. H., & Forrest, S. (1993). Relative building-block fitness and the building block hypothesis. D. Whitley, Foundations of Genetic Algorithms, 2, 109-126. (Sección 5).

[2] Schaffer, J.D., et al., A study of control parameters affecting online performance of genetic algorithms for function optimization, in 3rd International Conference on Genetic Algorithms. 1989: San Mateo, California. p. 51-60.