Projeto II - Buscas de Melhoria Iterativa e Satisfação de Restrição

CTC- 17 Inteligência Artificial

Prof. Paulo André Castro
Alyson Fernandes Basilio
25 de setembro de 2017

1.Objetivo

Exercitar e fixar conhecimentos adquiridos sobre Resolução de Provas através de Busca de Melhoria Iterativa (onde o destino é a solução, não o caminho) e sobre Problema de Satisfação de Restrições. A linguagem utilizada para a implementação dos algoritmos foi o Python.

2. Descrição Atividade 1 – Melhoria Iterativa

Crie um sistema capaz de resolver o problema das N-Rainhas através de busca de melhoria iterativa (hill climbing ou têmpera simulada) para N com os seguintes valores: 10, 15, 20 e 25. Tabele os tempos de processamento para obter uma solução.

2.1. Solução: Algoritmo Hill Climbing

Em primeiro lugar, definiu-se um estado aleatório inicial. A peculiaridade é que, nesse estado, não se encontram mais de uma rainha por linha ou por coluna. Logo, basta avaliarmos as diagonais. Então, definiu-se que a diferença entre um estado e outro seria a permutação de duas rainhas. Para a função de aptidão, foi utilizada a seguinte métrica: contou-se quantas rainhas se atacavam na diagonal. A solução era encontrada quando esse valor chegava a zero. A partir disso, seguiu-se o algoritmo descrito no pseudo-código a seguir:

```
# Numero de rainhas
n = 10
initialState = []
for i in range(n):
    q = randint(0,n-1)
    while q in initialState:
        q = randint(0, n - 1)
    initialState.append(q)
hillClimb(initialState)
print("Solucao:")
printTabuleiro()

def hillClimb(initialState):
    currentState = initialState
    aptidao = getAptidao()
    k = 0
    while aptidao != 1:
        initialState = currentState
        makeAMove()
```

```
        aTemp = getAptidao()
        if aTemp>aptidao:
            aptidao = aTemp
            currentState = initialState[i]
        k+=1
    print('Iteracoes:', k)
    return
```

Tabela 1: Resultados obtidos pela execução do algoritmo

	N=10	N=15	N=20	N=25
Tempo Médio de Execução(s)	1.05	271	-	-
Número de Iterações Médio	4430	538828	-	-

Como se pode ver na Tabela 1, o algoritmo não conseguiu resolver o problema para N = 20 e nem para N=25 por tempo de execução estourado, o aluno deixou o algoritmo rodando a noite inteira para os dois casos. Não foi encontrado o porque do algoritmo não conseguir encontrar o resultado para esses casos.

3. Descrição Atividade 2 – Melhoria Iterativa

Crie um sistema capaz de resolver o problema das N-Rainhas através de busca de melhoria iterativa (hill climbing ou têmpera simulada) para N com os seguintes valores: 10, 15, 20 e 25. Tabele os tempos de processamento para obter uma solução.Usando um algoritmo distinto do item 2.1, determine o máximo global da função abaixo. (Resoluções analíticas não são aceitáveis). Você encontrou algum ponto de máximo local ? Qual(is)?

$$f(x,y) = 4e^{-(x^2+y^2)} + e^{-((x-5)^2+(y-5)^2)} + e^{-((x+5)^2+(y-5)^2)} + e^{-((x-5)^2+(y+5)^2)} + e^{-((x+5)^2+(y+5)^2)}$$

3.1. Solução: Algoritmo Tempera Simulada

Em primeiro lugar, definiu-se um estado aleatório inicial, um par de valores reais (x,y) no intervalo [-10,10]. A partir disso, aplicou-se o algoritmo, como se mostra no código a seguir:

```
temp = 1000000
coolingRate = 0.0003
k=0
while temp > 1:
    currentState = [random.random() * randint(int(initialState[0]) - 10,
                   int(initialState[0]) + 10), random.random() *
                   randint(int(initialState[1]) - 10, int(initialState[1]) +
                   10)]
    D_e = getAptidao(currentState) - getAptidao(initialState)
    if D_e>0:
        initialState[0]=currentState[0]
        initialState[1]=currentState[1]
```

```
    else:
        if math.exp(D_e/temp)<random.random():
            initialState[0] = currentState[0]
            initialState[1] = currentState[1]
    temp *= 1-coolingRate
    k+=1
print('Iteracoes:', k)
print('Max encontrado:',getAptidao(initialState))
resultados.append(getAptidao(initialState))
iteracoes.append(k)
return
```

3.2. Resultados e Discussões

Rodando o algoritmo 100 vezes para entradas aleatoriamente escolhidas o maior valor encontrado foi X, e o tempo médio de solução foi de Y.

4. Conclusões

Foi interessante aprender soluções de inteligência para resolução de Provas através de melhoria iterativa.

Na primeira atividade o aluno não teve dificuldades para a resolução do problema.

O aluno não conseguiu completar a segunda atividade a tempo de entregar este relatório.