MP rep5　　205713B 朝比奈太郎　2022/07/20

(練習問題 A)

４人の院生A,B,C,Dが、英語、数学、物理、化学の試験
問題作成を分担する。４人はそれぞれ専門が違うので、
次表のように問題作成に作業時間が必要である。

	英語	数学	物理	化学
A	6	1	9	3
B	2	5	7	8
C	6	3	5	4
D	3	5	2	1

上記の割り当て問題において英語=1, 数学=2, 物理=3, 化学=4 で表す。この時の 1 つの可能解 X が(A,B,C,D)=(3,4,1,2)である場合、近傍探索法を適用するとどうなるか。

1)
(3,4,1,2)に摂動を適応すると。x_1=(4,1,2,3)となる。
このとき目的関数値は、f(x_1)=3+2+3+2=10

2)
(3,4,1,2)に摂動を適応すると、x_2=(1,2,3,4)となる。
このとき目的関数値は、f(x_2)=6+5+5+1=17

3)
f(x_1)=10, f(x_2)=17 であり、この問題の最適解は f(X)=9 である。従って、f(x)=9 となる摂動を考える。
最適解のとき、すなわち f(x)=9 のとき、(2,1,3,4)をとるのでこれを満たす摂動を適宜すれば良い。
従って、(3 -> 2, 4 -> 1, 1 -> 3, 2 -> 4)という処理を摂動として定義する。
適用後、f(x)= 1+2+5+1=9,
これは 1),2)よりも良い解(f(X)の値が小さい)といえる。

(練習問題 B)

1)

```
#初期解を与える
```

```
#解が改善されなくなるまで解の更新を繰り返す
#指定した終了条件が満たされるまで上記を繰り返す。

A = [6,1,9,3]
B = [2,5,7,8]
C = [6,3,5,4]
D = [3,5,2,1]

wariate = [A,B,C,D]
#print(wariate)
P = [[1,2,3,4], [1,3,2,4], [1,3,4,2],[1,4,2,3],[1,4,3,2]]  #初期解
perturbation = [[3,4,1,2],[2,4,1,3], [4,2,1,3],[2,3,1,4],[3,2,1,4]] #摂動後
def improve(x):
    #f(x)を求めるアルゴリズムを記述
    #初期解が P 個与えられたとき、P 回繰り返す
    sum = 0
    i = 0
    for num in x:
        sum += wariate[i][num-1]
        i += 1

    return sum

def multi_start_local_search():
    count = 0
    for i in range(len(P)):
        initial_solution = P[i]
        #while True:#終了条件が満たされるまで繰り返す -> 終了条件更新は P 回まで
        x = improve(initial_solution) #初期解を与える
        y = improve(perturbation[i])
        #x_ = x or y #最良の解を暫定解とする
        #f(X)>f(y)を満たす近傍解 y が存在すれば新しい暫定解とする。
        if(y > x): #x と y どちらが良いか判定
            x_ = x
        else:
            x_ = y
```

```
    return x_
```

2)
摂動 -> 「数字を 1 つ隣に rotate する」
停止条件 -> 終了条件更新 P 回まで

上のプログラムを実行結果
初期解: (1,2,3,4)のとき 17(集団内での最良解)
　　(3,4,1,2)のとき 28(集団内での最悪解)
最終解: (4,1,2,3)のとき 10(集団内での最良解)
　　(2,3,4,1)のとき 15(集団内での最悪解)

4)
より良い解を得るための初期解の選び方
まず、良い解を得るためには、最適解に近い解または、最適解を見つける必要がある。従って、暫定解の範囲を広くするために初期解をたくさん設定するべきであると考える。(可能領域内において)
2)では、少ない初期解(P=2)でたまたま最適解が見つかったが最適解をわざと外した上で、初期解を多く設定し、より良い解が得られるか検証する。

摂動「数字を 2 つ隣に rotate する」
初期解が 5 つのとき
初期解 -> [[1,2,3,4], [1,3,2,4], [1,3,4,2],[1,4,2,3],[1,4,3,2]]
摂動後 -> [[3,4,1,2],[2,4,1,3], [4,2,1,3],[2,3,1,4],[3,2,1,4]]
最良解 15

初期解が 3 つのとき
初期解 -> [[1,2,3,4], [1,3,2,4], [1,3,4,2]]
摂動後 -> [[3,4,1,2],[2,4,1,3], [4,2,1,3]]
最良解 16
初期解が 1 つのとき
初期解 -> [1,2,3,4]
摂動後 -> [3,4,1,2]
最良解 17

上記より、初期解が 5 個のとき、3 このとき、1 つのときの順番で最良解が最適解に近かっ

たことから、最適解に近い最良解を求めるためには初期解の数を増やすことが有効であると言える。また、アルゴリズムの特性上、仮に最適解を見つけたとしてもそれ以上更新されることなく、最適解で止まることから初期解の数が多いに越したことはないと言える。