2010 年度 大問 5

hari64boli64 (hari64boli64@gmail.com)

2023 年 5 月 3 日

1 問題

$$\max \sum_{i=1}^{n} p_i x_i$$
$$\text{subject to} \sum_{i=1}^{n} s_i x_i \leq S \quad (*)$$
$$x_i \in \{0,1\} \ (i = 1, \cdots, n) \quad (*)$$

2 解答

5.py が解答。

slow とあるのは、正当性の検証目的であり、実際には不要。

ソースコード 1　answer

```
import numpy as np
import random
from itertools import product


def makeProblem(problem: str):
    if problem == "P":
        N = 5
        S = 5
        Ps = [2, 3, 2, 1, 3]
        Ss = [2, 3, 1, 2, 1]
        return N, S, Ps, Ss
    elif problem == "Q":
        N = random.randint(1, 5)
```

```
        S = random.randint(1, 10)
        Ps = [random.randint(1, 10) for _ in range(N)]
        Ss = [random.randint(1, 10) for _ in range(N)]
        return N, S, Ps, Ss
    elif problem == "R":
        N = random.randint(1, 5)
        S = random.randint(1, 10)
        W = random.randint(1, 10)
        Ps = [random.randint(1, 10) for _ in range(N)]
        Ss = [random.randint(1, 10) for _ in range(N)]
        Ws = [random.randint(1, 10) for _ in range(N)]
        return N, S, W, Ps, Ss, Ws
    else:
        raise ValueError("problem must be P, Q or R")


def slowP(N, S, Ps, Ss):
    """
    bit全探索による(P)の解法

    解法を説明すると、
    1. 0~2^N-1までのbitを生成する これは、各x_iが1かどうかを表す
    2. そのbitに対応するxについての制約条件を計算する
    3. 制約条件を満たすならば、その
       bitに対応するxについての目的関数の値を計算する
    4. 3.で計算した値の最大値を答えとする
    というものである。

    これは、bit全探索の計算量がO(2^N)であり、指数時間アルゴリズム
       となっている。
    """
    ans = -np.inf
    for bit in range(1 << N):
        constraint = sum([Ss[i] * bool(bit & (1 << i)) for i in
           range(N)])
        if constraint <= S:
            obj = sum([Ps[i] * bool(bit & (1 << i)) for i in
               range(N)])
            ans = max(ans, obj)
    return ans


def slowQ(N, S, Ps, Ss):
    """
    全探索による(Q)の解法

    上とほぼ同様
```

```
58      """
59      ans = -np.inf
60      for Xs in product(range(0, 10 + 1), repeat=N):
61          constraint = sum([Ss[i] * Xs[i] for i in range(N)])
62          if constraint <= S:
63              obj = sum([Ps[i] * Xs[i] for i in range(N)])
64              ans = max(ans, obj)
65      return ans
66
67
68  def slowR(N, S, W, Ps, Ss, Ws):
69      """
70      全探索による(R)の解法
71
72      上とほぼ同様
73      """
74      ans = -np.inf
75      for bit in range(1 << N):
76          constraint1 = sum([Ss[i] * bool(bit & (1 << i)) for i in
                range(N)])
77          constraint2 = sum([Ws[i] * bool(bit & (1 << i)) for i in
                range(N)])
78          if constraint1 <= S and constraint2 <= W:
79              obj = sum([Ps[i] * bool(bit & (1 << i)) for i in
                    range(N)])
80              ans = max(ans, obj)
81      return ans
82
83
84  def solveP():
85      """
86      (1),(2)の解答
87      これは p1,…,pn, s1,…,sn について多項式時間, S について指数
            時間アルゴリズムである。
88      """
89      N, S, Ps, Ss = makeProblem("P")
90      print(f"{N=},{S=},{Ps=},{Ss=}")
91      print(f"{slowP(N, S, Ps, Ss)=}")
92
93      As = [[None for _ in range(S + 1)] for _ in range(N + 1)]
94      for s in range(1, S + 1):
95          As[0][s] = -np.inf
96      As[0][0] = 0
97
98      for j in range(1, N + 1):
99          for s in range(S + 1):
100             if s < Ss[j - 1]:
```

```
                    As[j][s] = As[j - 1][s]
                else:
                    As[j][s] = max(As[j - 1][s], Ps[j - 1] + As[j -
                        1][s - Ss[j - 1]])

    print("As=", *As, sep="\n")
    print("ans=", max(max(a) for a in As))


def solveQ():
    """
    (3)の解答
    これは p1,…,pn, s1,…,sn について多項式時間, S について指数
        時間アルゴリズムである。
    """
    N, S, Ps, Ss = makeProblem("Q")
    print(f"{N=},{S=},{Ps=},{Ss=}")
    print(f"{slowQ(N, S, Ps, Ss)=}")

    As = [[None for _ in range(S + 1)] for _ in range(N + 1)]
    for s in range(1, S + 1):
        As[0][s] = -np.inf
    As[0][0] = 0

    for j in range(1, N + 1):
        for s in range(S + 1):
            # ここが漸化式
            if s < Ss[j - 1]:
                As[j][s] = As[j - 1][s]
            else:
                As[j][s] = max(
                    As[j - 1][s],
                    # 以下が増えた これは、x_iを1~10の範囲内で変
                        化させている
                    max(
                        Ps[j - 1] * x + As[j - 1][s - Ss[j - 1] *
                            x]
                        for x in range(1, 10 + 1)
                        if s - Ss[j - 1] * x >= 0
                    ),
                )

    # print("As=", *As, sep="\n")
    print("ans=", max(max(a) for a in As))


def solveR():
```

```
    """
    (4)の解答
    これは p1,…,pn, s1,…,sn, w1,…,wn について多項式時間, S について指数時間アルゴリズムである。
    """
    N, S, W, Ps, Ss, Ws = makeProblem("R")
    print(f"{N=},{S=},{W=},{Ps=},{Ss=},{Ws=}")
    print(f"{slowR(N, S, W, Ps, Ss, Ws)=}")

    # Aの引数にWを追加すれば良い
    As = [[[None for _ in range(W + 1)] for _ in range(S + 1)]
          for _ in range(N + 1)]
    for s in range(S + 1):
        for w in range(W + 1):
            if s != 0 or w != 0:
                As[0][s][w] = -np.inf
    As[0][0][0] = 0

    for j in range(1, N + 1):
        for s in range(S + 1):
            for w in range(W + 1):
                if s < Ss[j - 1] or w < Ws[j - 1]:
                    As[j][s][w] = As[j - 1][s][w]
                else:
                    As[j][s][w] = max(
                        As[j - 1][s][w],
                        Ps[j - 1] + As[j - 1][s - Ss[j - 1]][w -
                            Ws[j - 1]],
                    )

    # print("As=", *As, sep="\n")
    print("ans=", max(max(max(a) for a in aa) for aa in As))


def main():
    solveP()
    print("=" * 10)
    solveQ()
    print("=" * 10)
    solveR()


if __name__ == "__main__":
    main()
```

ソースコード 2　output

```
N=5,S=5,Ps=[2, 3, 2, 1, 3],Ss=[2, 3, 1, 2, 1]
```

```
slowP(N, S, Ps, Ss)=8
As=
[0, -inf, -inf, -inf, -inf, -inf]
[0, -inf, 2, -inf, -inf, -inf]
[0, -inf, 2, 3, -inf, 5]
[0, 2, 2, 4, 5, 5]
[0, 2, 2, 4, 5, 5]
[0, 3, 5, 5, 7, 8]
ans= 8
==========
N=3,S=3,Ps=[7, 10, 4],Ss=[1, 4, 1]
slowQ(N, S, Ps, Ss)=21
ans= 21
==========
N=4,S=2,W=8,Ps=[5, 7, 7, 10],Ss=[2, 8, 7, 2],Ws=[6, 2, 6, 10]
slowR(N, S, W, Ps, Ss, Ws)=5
ans= 5
```

3 知識

特に無し